Dr. T's Programmable Variations Generator and Master Editor

by Emile Tobenfeld

Manual by Jim Aikin

Additional material by Emile Tobenfeld and Jim Johnson

Copyright 1988
All Rights Reserved

CONTENTS

PUBLISHER'S NOTE 1

PREFACE 3

CHAPTER 1 **Understanding the PVG** 7

CHAPTER 2 **A Quick Overview of PVG Operations** 11

Presets 12

Macros 13

The PVG Screens 13

CHAPTER 3 **General Options** 19

Overwrite Original 19

Consecutive Mults & Evolving Mults 19

Reverse 20

Changes per Vary 21

Edit Mode 21

OK, CLEAR, CANCEL, & RANGE 22

Preset Name 23

CHAPTER 4 **Protection** **25**

Probability Of Protection 26

Scale Positions 26

R (Reverse) 26

First & Last 27

Chn 27

Pit 27

Vel, Dur, & Tim 28

Int & VInt 28

NTim, NInt, & NVIn 29

CHAPTER 5 **Restrictions** **31**

Scale Positions 31

Pitch Limits 32

Velocity Limits 32

Minimum Time 32

Autocorrect 33

Duration Adjust & Duration Limit 34

CHAPTER 6 **Defaults** **35**

Limit Pitch Range 35

Limit Velocity Range 35

SD Default Factor 35

Allow New Scale Positions 36

CHAPTER 7 **Changes** 37
AMT & WGT 37
Constant, Gaussian, & Signed Changes 39
Pitch, Velocity, & Duration 40
Time & Shift 40
Staccato/Legato 42
Interval 42

CHAPTER 8 **Swap/Copy** 43
SWAP, ADJ, & COPY 43
PROCESS CONSECUTIVE 45
Copy 45
Copy From Seq 46
Time Reverse 46
Rotate 46
Xtime 47
Copy Loop 48
Set Values 49
Interval 50
Delete & Erase 50

CHAPTER 10 **Global Changes** 53
Global 1 54
TRANSPOSES & INVERSIONS 54
Time Reverse 55
Delete & Erase 55
Global 2 55
SET VALUES 56
ROTATE VALUES 56
TIME REV 57

CHAPTER 11 **Split/Pattern** **59**
E (End) 60
Note Buttons 60
Interval Data Fields 60
Arithmetic Tests 61
Use if Match & Use if no Match 61
Use Whole Pattern 62
Extend 62
Velocity Intervals 63
SPLIT & Delete Split Notes 63

CHAPTER 12 **In Betweens** **65**
Sequence 65
Number of Sets 65
Autocorrect 66
Consecutive Mults 66
Restrict Scale Positions 66

CHAPTER 13 **Ornaments** **67**
Offset & Delay 68
Pitch, Velocity, Duration, & Channel 69
Cut 70
COPY 70
Loop, Length, & Shift 70
Mod & Max 72
Split 72
Extend 72
Next Note Lim, Duration Lim, & Abs 73

CHAPTER 14 **Add Controllers** **75**

Delay, Type, & Value 75

Pgm & Offset 77

Loop & Length 77

COPY 77

Split 78

Next Note Lim, Extend, & Duration 78

Modulation 78

CHAPTER 15 **Vary Controllers** **83**

Protection & Restrictions 83

Global Processes 84

Single Processes 85

CHAPTER 16 **Get, Store, Load, & Save** **87**

CHAPTER 17 **Macros** **89**

The Preset List 90

General Options 91

Lines 92

Con Lines 92

Range Unit 93

Ops 93

From/To Ranges & Cycles 93

CHAPTER 18 **Master Editor** 99

Edit Range 99

Blend 100

Chords 103

Controllers 106

Tempo Changes 107

Track Utilities 108

Pitch Map 109

CHAPTER 19 **Polyphonic Key Pressure** 113

INDEX 115

Publisher's Note About This Manual

Thank you for purchasing the Level II version of the Keyboard Controlled Sequencer. This manual describes the features contained in Level II that are not present in Version 1.6 of the KCS. If your disk contains a file named README, be sure to read it before using the program, as it may contain information about changes made since the completion of this manual.

The Level II KCS contains two extremely powerful editing modules that are not present in V1.6: the Programmable Variations Generator (PVG) and the Master Editor. These modules are closely related, as most of the features of the PVG can be used for editing as well as the generation of variations. Level II also supports the recording, playback, and editing of polyphonic key pressure events.

The first 17 chapters of this manual describe the operation of the PVG, while Chapter 18 describes the Master Editor and Chapter 19 describes the handling of polyphonic key pressure events.

A Note on the PVG

While the Programmable Variations Generator has been subjected to extensive testing by many musicians during the course of it's development, there is no way we could possibly test all the possible combinations of variation operations, parameter settings, and input sequences. Due to the nature of the program, it is not possible to predict the results you will obtain with any degree of certainty, except in the case of strictly deterministic (non-random) editing. For this reason, please remember that you may not always be able to get exactly what you're looking for from the program.

The PVG is an enormously capable musical instrument, and is arguably the most powerful composition system ever seen on this planet. We at Dr. T's have only begun to scratch the surface of the system's capabilities. While we are happy to provide customer support for situations involving catastrophic bugs and other major problems, we are not, at this time, able to provide in-depth applications support for the PVG.

Even though we have not experienced any severe problems with the PVG, the experimental nature of the program makes it wise to back up your sequence data more frequently than you might normally.

Preface

For John Cage, who opened all of our ears to new paradigms in music making, and Tom Trobaugh (1955-1987) whose creative energies inspired all those who knew him.

History

In January of 1986, we published our Commodore 64 Algorithmic Composer package, three programs for the algorithmic generation of music. An immediate reaction of some users, including myself, was that it would be nice to be able to algorithmically modify music that we had recorded ourselves. I started thinking about the possibilities for a program called Algorithmic Variations, and attempted to cajole Jim Johnson, one of the authors of the Algorithmic Composer, to write it. He decided, correctly in my opinion, that it would be very difficult to write such a program on the Commodore and make it run smoothly enough to be worth using, and that the market for such a program probably would not justify the effort.

While working on the KCS for the C128 and ST, I did some thinking about the things that one would want in such a program. In January and February of 1987, I took a 'vacation' from working on the ST KCS, and implemented many of the features found in the **Changes** and **Swap/Copy** screens of the PVG. I found the program fascinating to use, and made some interesting music. I found that I could be almost hypnotized by a series of evolving variations that a simple preset could generate from a simple line of maybe a dozen notes, and even more fascinated if I played two or three sets of such variations slightly out of phase with each other.

As I made additions to the PVG, the nature of the program changed somewhat. It was originally intended as a tool to allow these changes to evolve over time. Although the user had very precise control over the *kinds* of changes to be made, the process was essentially random in nature. As I added features like global changes and macros, it became possible to use the program to do quite complex things that were completely determined. These can either create variations on a piece or just be editing operations. The PVG in fact provides a very powerful editing environment with a number of useful orchestration tools as well.

The Master Editor contains additional edit operations that did not fit into the PVG scheme. Actually, many of the features can be used either for the generation of variations or for editing, which makes them as much part of the Master Editor as part of the PVG.

A consistent part of my approach to designing and coding the PVG has been the desire to make things as general as possible. If I let you do something to the pitch of a note, I am likely to let you do the same thing to its velocity, duration, timing, MIDI channel, perhaps all at once. Likewise, in addition to allowing you to change, set, copy or swap parameter values, I also allow you to do things like rotate and time-reverse. This leads to a program with many screens, and more possibilities than one can absorb in a few hours or even days. Fortunately, the program is organized so that you can usually ignore those features that you do not want to use.

Using the PVG

I suggest that you bite off this program in small chunks, trying a few features here and there and seeing where they take you musically, then trying some more, etc. If you need to do something specific to a piece, search through the screens and the manual to see if there is a feature that will do the job. As you use the program, you should get a better picture of the kinds of things it can do, and how it various parts fit together.

The PVG and Master Editor can be used in a number of different ways. You can use them strictly as editing tools to put together a preconceived piece of music as rapidly and accurately as possible. You can use them as a grab-bag of tools, available to be applied to a musical phrase any time you are so moved by whim or inspiration. They can also be explored as a source of music in themselves. You can, for example, set up a complex macro (random or not) using the consecutive lines feature, and try applying it to different input phrases and compare the different output variations that are produced.

As of this writing (December 1987) I have been reading James Gleick's excellent book "Chaos Making a New Science". This got me thinking about using the Level II KCS to generate musical analogs to some of the processes described in the book, which contains some beautiful computer graphics images generated from things like Mandelbrot diagrams.

You may ask "What is so musically appealing about chaos?", and this is certainly a reasonable question. As it happens, the physical processes that involve chaos include those responsible for determining the shapes of things like clouds, waterfalls, tree branches, and leaves; all of which are commonly thought to be beautiful and appropriate subjects for visual artists in all media.

I realized that there were many ways to accomplish this using macros in the PVG, but I also added the recursive feature in the Pitch Map section of the Master Editor. This allows you to take a sequence, apply the pitch map to it, append the result to the original, apply the same pitch map to the result, append again ... The results I obtained in a couple hours of experimentation were quite promising.

Another interesting trick is to take a sequence of only a single note (perhaps including a controller value or two), and create a macro with a series of lines that can modify it. Set the **Lines** value to 1, and make a large number of variations. The results are *far* more interesting than you would expect from the description, especially if two or three parts generated with the same macro are played out of phase. Alternatively, you can use protection to cause each line to operate only on certain notes, set the **Lines** value to 0, and create a non-randomly evolving line from the single note.

Of course, a photographer does not just point his camera at a random tree, cloud or waterfall and snap the shutter. To obtain a good picture, he must consider lighting, angle, framing and other compositional aspects. Likewise, a composer using a tool like the PVG to generate musical lines must consider compositional aspects such has overall structure, sound choices, and the textures obtained by combining different parts to create a successful composition. Even though many of the musical lines in a piece were generated by a computer program, the composition is as much a creation of the composer as a good photograph is the creation of the photographer.

One aspect of the graphics obtained from chaos theory is a property called unity (or symmetry) over scale. Small pieces of the drawing when blown up look much like the entire drawing. There are a number of ways to simulate this using the KCS, I leave them to your imagination.

Acknowledgement

I would like to acknowledge the support and suggestions of more friends and users than I can comfortably list here. The Chords feature of the Master Editor was suggested by Jeff Baker, the Auto-Correct and Follow features are extensions of ideas proposed by Jeff Baker and Iasos. The Match features are extensions of ideas first developed by Jim Johnson in his Series program for the C64. The Pitch Map feature is an extension of ideas first suggested by Richard Munson. The Macro feature was suggested by Tom Trobaugh and Laurie Spiegel, and Pattern Matching was suggested by Tom Trobaugh. I would also like to thank Al Hospers, Bruce Carroll, Bob Rees and George Lewis for encouragement and helpful discussions.

Emile Tobenfeld
(Dr. T)

Chapter 1

Understanding the PVG

What do you want to do with your music? No, I don't mean once it's finished. Dr. T can't help you get a record contract, or put together a performing group. I mean, what do you want to do with your music as you're in the process of creating it?

The simplest way to use your Keyboard Controlled Sequencer is as a real-time MIDI recording and playback tool. Once you've recorded some tracks or sequences, though, you'll probably find yourself using the many editing features of the KCS to make alterations, small or large, in the data. That's the wonderful thing about MIDI sequencing: It turns music into an open-ended, plastic medium. In a powerful sequencer like the KCS, anything from fixing an isolated mistake to restructuring an entire piece can be handled with a few quick keystrokes.

Even without the Programmable Variations Generator, the KCS contains just about all of the standard editing tools found in the current generation of MIDI sequencers, not to mention a handful of tools that aren't found on any other sequencer. And for many types of music, standard editing may be all you'll need. Yet the PVG goes much further. It is a much more generalized environment for making changes in music data - a tool both for editing music that already exists, and for generating entirely new music. After using it for a while, you may even find yourself envisioning whole new compositional structures that can take advantage of the PVG's unique strengths.

As an editing tool, the PVG lets you make a wide variety of precisely controlled changes in a track or sequence. Some of these actually duplicate the functions found in the standard KCS editing environment or the Master Editor. But there's a reason for the duplications, as you'll see when you start building macros. Other changes, while still precisely controlled in nature, go further. For example, you could transpose a sequence up by a half-step while instructing the PVG to ignore some specific notes and transpose others. If it ignores C, D, F, and G while transposing Eb, Ab, and Bb, the result of the operation would be to modulate a sequence in the key of C minor into the key of C major. Needless to say, the PVG can do edits far more complex than this.

If this were all it did, however, there would be no reason to call it a "variations generator". A key concept behind the program is that it allows you to make changes in a track or sequence while specifying some aspects of the change and letting the computer determine other aspects for you. Generally speaking, the user defines the changes to be made in broad terms, and then the computer fills in the details at random. For example, you might want to make a number of small variations in note velocity in order to make a melody line sound a bit less stiff. Instead of typing in a number of values on the KCS edit screen, which could take some time, you can instruct the PVG to alter the velocities for you. This is rather like a painter blotting an area of a watercolor with a terrycloth towel to add texture. The nature of the desired effect is known in advance, and the handling of the towel may require some skill; yet the precise pattern that will emerge within any given square centimeter is absolutely not under the control of the painter. (And if the painter botches the job, the watercolor may be ruined. With the PVG, you can simply click on **Undo** and try again!)

Even this example only hints at the power of the PVG. At the deepest level, it is a resource for machine-generated composition. This rather specialized field has existed only since the early 1950s, when the primitive computers of the day were first used to choose pitches and durations for instrumentalists to perform.

At about the same time, John Cage and other composers began experimenting with indeterminacy in composition using random number tables, star charts, and other types of data that had not previously been considered musical resources. The interest in indeterminacy was primarily a revolt against the rigidly authoritarian system of total serialism in composition, which was very much in vogue in college and university music departments at the time. Indeterminacy also struck a blow against expressionism, the belief that music ought to express or evoke human emotions. At its best, indeterminacy in composition embodies an Oriental philosophy of total acceptance of the present moment.

Another term for indeterminate music is 'aleatoric' music, from the Latin word 'alea', which means dice. (Okay, the Latin scholars who raised their hands get a gold star. Actually, 'alea' means die. The plural is 'aleae.') And above all, the Programmable Variations Generator is a way of playing dice with your music. You may consider the results expressive, or not expressive, and you may want to mix chance with choice in various ways, but the dice-throwing operation is the same in any case.

You may not always like the results of the PVG's operations, or find them musically useful. Like any musical tool, this software requires some effort to learn, and still more effort to use effectively. Musical tools also require an esthetic context in which they can have meaning. It's instructive to remember, for example, that when Mozart sat down at the piano, all of the chords that Dave Brubeck uses were right there under his fingers. He couldn't find them, however. If he did accidentally stumble onto one, it sounded like wrong notes. The reason is that the isolated jazz chord had no context. The harmonic language that would have made it meaningful was not yet born.

In the same way, the results of PVG operations may sometimes sound like gibberish, not because they are intrinsically bad or useless but simply because the software tool is capable of far more than we can understand, given our own present-day esthetic perspectives. Unlike real-time interactive "improvising" software, the PVG is not optimized to produce music that is immediately pleasing to the average listener. It requires a little more effort to use well, because it is capable of far more musical variety.

What the PVG does is allow a composer to try out many compositional processes rather quickly, in order to find the ones that sound best in a particular context, or that suggest some new context that can be made meaningful. Instead of constructing elaborate sequences with time-consuming single-step data entry, the composer can use high-level processes to generate large amounts of material quickly, seize the portions of the material that sound good, and move on.

Chapter 2

A Quick Overview of PVG Operations

When you click on **PVG** in the KCS edit screen, you will find yourself looking at the first of a number of screens that are part of the PVG. You use the PVG by clicking on one or more of the data fields that appear on the screen and then typing numbers from your computer keyboard or incrementing them by clicking the mouse on the arithmetic strip in the lower left corner of the screen. After entering whatever data you need for your current operation (or recalling a preset that contains the appropriate data), simply click on **OK** in the lower right-hand corner of the screen, and the PVG will do its thing. When it's finished, it will return you to the edit screen, which now displays the variation that has just been generated, or the last variation in a series of new ones.

If you decide not to create a variation, click on **CANCEL** instead of **OK**, and no variation will be generated. **NOTE:** If the PVG is unable to perform your operation, an error message (such as "Track or sequence contains no unprotected notes!") will appear at the bottom of the screen. In this situation, clicking on **CANCEL** will have no effect. Don't panic. Simply press any key on the computer keyboard, and make whatever changes are necessary.

The PVG will always perform its operations upon whatever track or sequence was in the edit screen just before you clicked on **PVG** (or upon a region that you have selected within that track or sequence). All of the operations of the PVG (except in-betweens) can be performed on either a track in Track mode or a sequence in Open mode. In this manual, to avoid confusion, we will refer to source and destination material as sequences.

Presets

The most basic way of using the PVG is by entering values in the various fields each time you want to create a variation. If you do that, you'll be treating it simply as a variations generator. The "programmable" part of the name should hint that there is more to the software, however. All of the values that you have entered can be saved together as a preset.

The PVG will hold up to 80 presets in memory at a time - 20 each for varying notes (the first six screens), creating ornaments, varying controllers, and adding controllers. You can think of PVG presets as being like the programmable preset sounds in a synthesizer. The most important difference is that in a synthesizer preset, you will normally want to specify data values for most or all of the parameters. A typical PVG preset, on the other hand, might contain data for only one or two of the hundreds of parameters, and few presets will be likely to contain data for more than a dozen.

There's no need to use the programmability of the PVG if you don't want to. With a couple of exceptions, all of the data entered in its active buffer (the data displayed when you call up its various screens) will be remembered by the program when you exit back to the edit screen, and will still be there the next time you click on **PVG**. If you want to get rid of the data to start building a preset from scratch, simply click on the **CLEAR** button, which appears in the lower right corner of most of the PVG screens. Clearing is highly recommended, by the way: Because a single Vary Notes preset in the PVG fills up a number of screens, any stray bit of data left hanging around on a screen that you're not using for your current operation can and will cause drastic unforeseen complications in the results of an operation.

Macros

Beyond the level of presets is a still higher level of operation called macros. A macro is basically an ordered list of up to 16 of the existing presets, along with a table of data telling where and how the presets will be used. This is the most complex and flexible editing environment found in the KCS (or in any other sequencer that we know of). Macros allow you to do things like change the velocity or transposition of all notes that fall on selected beats of the bar. The PVG will store 20 macros, just as it will store presets. A macro can contain presets of any type. A macro does not contain the information in the presets, however; it contains only a list of preset numbers, along with some other information. If you overwrite one of the presets in memory with a new one, you will change the effect of any macro that uses that preset.

The PVG is shipped with a selection of basic editing presets and associated macros. The current set of presets and macros can be stored to and loaded from disk.

The PVG Screens

If you haven't loaded the KCS, please do so now. In this section, we'll take a quick look at the various modules and features within the PVG. Then, in the following sections, we'll examine each of these modules and features in detail.

The first time you call up the PVG, you'll find yourself in the **Changes** screen. Across the upper half of the screen are rows and columns of data fields. These are used for making randomly placed changes in the musical elements listed along the left edge (**Pitch**, **Velocity**, **Duration**, **Time**, **Shift**, and **Interval**). Below this grid are two other areas, one labelled **RESTRICTIONS** and one labelled **GENERAL OPTIONS**.

As in the rest of the KCS, clicking on a word that's green will highlight it (black letters in a green rectangle). This is how options are switched on and off. In some cases, several adjacent words can be highlighted at the same time. In other cases, they may function like the buttons on a car radio: Pushing one in will cause another to pop up. We will use the terms "button" and "switch" to refer to these words.

The green word in the lower right-hand corner, however, is a bit different. It's the name of the current preset. Clicking on any letter of this will allow you to type in a new name.

When the visual cursor is on a data field (one of the numbers or empty horizontal underlines), the red arrows at the lower left, called the arithmetic strip, can be used with the mouse as an alternative to typing in values. Again, this system works just the way it does in the rest of the KCS. Clicking on one of the end diamonds will move the parameter to the end of its allowable range, clicking on one of the single arrows will move it by a single increment, and clicking on a double arrow will move it by some larger value, usually by tens.

Along the right side of the screen is a menu that lets you move from one screen to another. Try clicking on various items here to take a quick look at the other screens. You'll find that the **GENERAL OPTIONS** area is found in all of the screens that are used for editing presets. In most of the screens, the **RESTRICTIONS** area is replaced by a similar area called **PROTECTION**. In the **Vary Controllers** screen, both **PROTECTION** and **RESTRICTIONS** appear in an abbreviated form.

While the entire screen blacks out and is redrawn when you move from one menu selection to another, it's important to understand that there is only one set of data under **PROTECTION** and one set under **GENERAL OPTIONS**. For example, whatever data you enter under **PROTECTION** while on the **Swap/Copy** screen will still be displayed when you switch to **Set Values**. For that matter, the same data will still be there if you go off to the **Macros** screen or the KCS edit screen and then come back.

The changes made in your sequence in the **Changes** screen mostly have to do with shifting musical values up or down (or, in the case of time, forward or backward) by various amounts. In general, you will specify the amounts, and the PVG will randomly select which notes the changes will be applied to.

The **RESTRICTIONS** data fields are used to tell the computer what random values you don't want it to generate. For example, if you click on the black-key accidentals (A#, C#, D#, F#, and G#) in the note list, all pitch change operations will produce pitches within the white-key C major scale. (There are many operations that restrictions don't apply to, so don't be too surprised if you wind up with some black-key notes in your variation even though you're using the restrictions just mentioned.)

The **Defaults** dialog box, which appears when you click on **Defaults** in the menu, is used mainly to tell the program what data you would like loaded into the **RESTRICTIONS** area automatically when you enter the PVG.

The **PROTECTION** options look superficially similar to the **RESTRICTIONS**, but are somewhat different in their effects. Protection is used to specify notes in the sequence that you don't want the computer to alter under any circumstances. For example, if you click on the black-key notes, as before, the result of pitch change operations will be that notes can change from one white key to another, or from a white to a black key - but once a given note has become a black key, it can't change any more. **NOTE:** In general, protection applies to an entire note, not to the individual data elements that make up the note. In other words, if your preset is randomly changing both velocities and pitches and you have specified protection for some pitches, once a given note has been transposed to one of the protected pitches, its velocity also becomes protected, and will not be changed by any subsequent randomizing operations of this preset.

The **GENERAL OPTIONS** data fields specify how many changes the PVG will make when it is activated, where the results will be stored, and a few other things.

The **Swap/Copy** screen is used for copying data from one part of the sequence onto another part, or for trading (swapping) data between two parts. You will specify the type and amount of data, and the PVG will determine exactly which notes in the sequence to copy or swap with which other notes.

In the **Set Values** screen, the computer is again deciding exactly which notes of the sequence to alter, but the nature of the result is being completely controlled. Instead of transposing the original pitch of the note up or down by some fixed or random amount, for example, which would be done on the **Changes** screen, in the **Set Values** screen the program would replace the original pitch with whatever specific pitch the user had entered.

The operations defined by the **Global 1 and Global 2** screens are applied not to single notes chosen by the computer, but to the entire contents of the sequence (except to whatever notes are protected by the **PROTECTION** parameters). Global operations are used more often in deterministic editing than in generating variations, but they can also be used quite effectively as elements in a complex variation process controlled by a macro.

The **Split/Pattern** screen is used for making complex logical tests. The **INTERVAL PATTERN SELECTION** section allows you to define pitch or velocity patterns of up to nine notes and tell the program which note within the pattern to vary. Pattern testing can be used in presets of the Vary Notes type; essentially, the testing is an extension of the tests set up in the **PROTECTION** area.

The **Ornaments** screen allows you to add new notes to a sequence. The new notes that are added will have some user-defined relationship to existing notes. The program can add new notes to randomly selected notes within a sequence, or to all unprotected notes. The term "ornaments" suggests turns, appoggiaturas, and trills, but a PVG ornament can be defined in other ways as well - for example, a chord of specified type can be added, or unison doubling on another MIDI channel.

The **Add Controllers** screen can be thought of as providing another type of ornamentation. Instead of generating new notes around an existing note, it generates controller messages, which might be used for pitch inflections or just about anything else that your synthesizer can do when it receives MIDI controller data.

The **Vary Controllers** screen is used to make changes in the existing controller events in a sequence. Changes made here can be either global (that is, applied to all the events of a given type) or single (applied to only one event at a time).

The **Macros** screen is used for assembling lists of presets to be used in a single PVG operation.

This page intentionally left blank.

Chapter 3

General Options

The basic idea of the Programmable Variations Generator is that it will make one or more changes in the contents of a track or sequence, and then store the results of its operations. Two questions come up immediately: First, how many changes will the PVG make in a given operation? And second, where will the results be stored? These questions are dealt with in the **GENERAL OPTIONS** section of the screen, which is part of all nine screens (**Changes**, **Swap/Copy**, **Set Values**, **Global 1**, **Global 2**, **Split/Pattern**, **Ornaments**, **Add Controllers**, and **Vary Controllers**) that are used in creating presets.

Overwrite Original

If you want to replace your sequence with the new material created by the PVG, click on **Overwrite Original** so that it is highlighted in inverse video. (Don't worry - if you don't like the new material, you can always click on **Undo** in the KCS edit screen and retrieve the previous version.) **Overwrite Original** will be used most often when doing edits with the PVG, as there's usually no need to keep the unedited version of a sequence lying around. When creating variations, you'll more likely want to keep the original, at least temporarily, as a reference and as a source for further variations.

Consecutive Mults & Evolving Mults

The next two selections, **Consecutive Mults** and **Evolving Mults**, are used when the PVG is generating more than one variation on the starting material. The number of variations that it will make at a time is selected by entering a number under **Variations** on the second line of the **GENERAL OPTIONS** section.

When **Consecutive Mults** (i.e., multiples) is highlighted, all of the variations will be stored consecutively on a single track. Be sure to check the time of the final DE event in your sequence before using **Consecutive Mults**, and set it to some appropriate value (such as the downbeat of the following bar). The reason for doing this is that the first event in each consecutive variation will start with reference to the time of the last DE event in the previous variation. Normally, a final DE event that's set seven or eight bars late because that's when you got around to stopping the recording process won't make any difference in the music, but in this case it will.

When you click on **Evolving Mults**, a very interesting thing happens. Without **Evolving Mults**, each variation created by the PVG in a given operation will be based on your original sequence. The **Evolving Mults** option causes the PVG to base each new variation on the preceding variation.

What effect this will have depends, obviously, on what the preset is doing. If it is setting every note to a fixed pitch, using **Evolving Mults** would be ridiculous, because once the notes have been set to this pitch, no further evolution can take place. When used with the randomizing options in the **Changes** and **Swap/Copy** sections, though, **Evolving Mults** can generate some wonderful patterns, where each new note or accent created by the computer becomes part of subsequent phrases. A single five-note or six-note phrase, subjected to various procedures in conjunction with **Evolving Mults**, could easily become the raw material for a long and convoluted piece of music.

Note that if you specify **Consecutive Mults**, the program will append the variation(s) to a copy of the original sequence, unless you are overwriting the original and doing a single variation.

Reverse

If the **Reverse** button is highlighted along with **Consecutive Mults and Evolving Mults**, the variations will be stored in the opposite order from the order in which they were created: The last-created variation will be at the beginning of the sequence, and the original version will be at the end. (The order of the individual notes within each variation will not be reversed, however.)

Changes per Vary

The number entered under **Changes per Vary** controls the number of operations the program will perform in the process of creating each new variation. This number interacts with the numbers entered in the **WGT** (weight) columns. Only changes for which you have entered non-zero weight values will be selected by the program. For example, if the only types of changes that have non-zero weights are pitch changes by 12 units (half-steps) with a weight of 2 and velocity changes by 24 units with a weight of 1, then each change will have a 2/3 chance of being a 12-unit (one octave) pitch change and a 1/3 chance of being a 24-unit velocity change. This will be explained more fully in Chapter 7, under the heading **AMT & WGT**.

If you select 0 under **Changes per Vary**, you will not get zero changes. Instead, the numbers under **WGT** will directly determine the number of changes made. The program will make one change for each weight unit. In the above case, this would mean three changes per variation--two one-octave pitch changes and one velocity change.

Edit Mode

The **Edit Mode** button is a quick way of setting up a preset so as to make a single set of changes in your original sequence. Selecting **Edit Mode** is equivalent to selecting **Overwrite Original** and entering 1 under **Variations**.

OK, CLEAR, CANCEL, & RANGE

Since the PVG's exit buttons are below the **GENERAL OPTIONS** section of the screen, we will discuss them here. Click on **OK** when you have selected or edited a preset and you are ready to have the machine generate its variation(s). Depending on how much material is being generated, you may be returned to the edit screen almost instantly, or the process may take a number of seconds. If you wish to abort the process, press the **ESC** key. (**Note:** If you click on **OK** without having selected any operations, you will see an error message at the bottom of the screen that says, "No weights set, cannot vary!" In this situation, you must exit the PVG by pressing a key on the computer keyboard.)

Use the **CANCEL** button if you want to go back to the edit screen without generating any variations. The **CLEAR** button is used for resetting all weights to zero. If you have been experimenting with the PVG's parameter settings during a session, we recommend that you use **CLEAR** before starting to set up a new preset. If you neglect this step, you may easily leave a parameter somewhere with a non-zero weight, which can cause confusion, wasted time, and useless musical results.

The **RANGE** button will appear to the left of the **CLEAR** button if you have selected a range in the edit screen (highlighting it by clicking and dragging on the note list) before entering the PVG. Also, if you have selected a range, you will find that the events at the beginning and end of the sequence that fall outside this range are now protected in the **PROTECTION** section. This data in the **PROTECTION** section can still be edited if desired; the purpose of the **RANGE** button is to restore these values in the **PROTECTION** section if they have been altered, so you won't have to exit back to the edit screen to see what the range limits are.

Preset Name

To the right of the **Edit Mode** switch are the name and number of the current preset. (If you are entering the PVG for the first time in a session, you won't see a number. The number will appear when you select **Get** in the menu and choose an existing preset, or when you store your new preset in one of the memory locations.) When you click on any letter of the name, the visual cursor will move to this letter, allowing you to type in a new name. Names can be up to 16 characters long. Be sure to give a preset a new name before storing it. If you don't, you'll end up with several presets named Default and no easy way of telling which is which.

This page intentionally left blank.

Chapter 4

Protection

There will often be portions of a sequence that you don't want affected by the PVG. For example, you might want each variation to begin with the same set of notes and diverge thereafter. Or you might want to exempt the low notes, in order to keep a bass line intact while the chords above it are being varied. This is what the **PROTECTION** sub-screen is for. (If you want to protect events that fall in cyclic rhythmic relationships, you'll need to use a macro rather than the **PROTECTION** parameters.)

The **PROTECTION** section appears whenever you click on **Swap/Copy, Set Values, Global 1, Global 2, Split/Patrn, Ornaments,** or **Add Cntrls** in the menu. Protection is also active when the **Changes** screen is being used, even though the **PROTECTION** area is not visible. Protection would be pointless in the controller variation process, since it protects notes, not controller events.

Entire events are protected by the protection feature, not the individual data (note number, time, duration, channel, and velocity) that make up the event. If you're using the PVG to alter some pitches and some velocities in a single operation, and you have protected the F#s, then if a note happens to be transposed to F# during the course of an evolving mult, neither its pitch nor its velocity will be varied by later operations. The protection tests are applied to a note's current values (which may already have been modified in the course of creating a variation), not to its original values. Certain operations, such as interval changes in the **Changes** section, can move a note from a protected pitch to an unprotected pitch - after which it might be varied further by some later operation.

Probability Of Protection

The number just to the right of the word **PROTECTION** on the screen is a probability factor with a range from 0 to 10. When this is set to 0, the data in the **PROTECTION** section will have no effect, no matter how it is set. When the probability factor is 10, the events you choose to protect will absolutely be protected. When the factor is 5, protected events have a 50% chance of being protected. In other words, they will have 50% less chance of being chosen by a random operation than unprotected events.

The probability factor is not applied to the protection of events at the beginning and end of the sequence under **First** and **Last**. These events will be absolutely protected in all cases.

The probability factor can be used in conjunction with global changes to protect some notes (chosen at random) but not others. This essentially defeats the nature of global changes, forcing them to operate pretty much like the usual random changes.

Scale Positions

Clicking on any of the note names in the **PROTECTION** sub-screen, causing it to be highlighted in inverse video, protects all notes with that pitch. To clear the display, click on the **X** to the right of the note names.

R (Reverse)

To the right of each line in the **PROTECTION** section is a letter **R**. This is a switch. When you click on it, it reverses, or logically inverts, the effect of the corresponding line. For example, if you have selected MIDI channels 3 and 7 under **Chn**, notes that fall on channels 3 and 7 will be protected, while all other notes will be fair game for the PVG's processes. If the **R** to the right of **Chn** is highlighted, however, all notes except those on channels 3 and 7 will be protected.

First & Last

Enter here the number of events at the beginning and end of the sequence that you want to be exempted from the variation process. If you have highlighted a range of notes in the edit screen before entering the PVG, you will notice that the limits of the range appear initially in the **First** and **Last** data fields. If you have edited these numbers and would like to get them back, click on the **RANGE** button at the bottom of the screen. **Note:** While the PVG's Vary Note presets do not vary non-note events (such as program changes at the top of a sequence), these events will be counted in figuring out what range you want to vary.

Chn

Here you can specify one or two MIDI channels; events on these channels will be protected (or exempted from protection by clicking on **R** to reverse the operation). Note that while the rest of the parameters in this column specify ranges of values to be protected, the data entered beside **Chn** refers to specific numbered channels, not a range of channels. If you would like to protect notes on a number of channels, use the **R** switch.

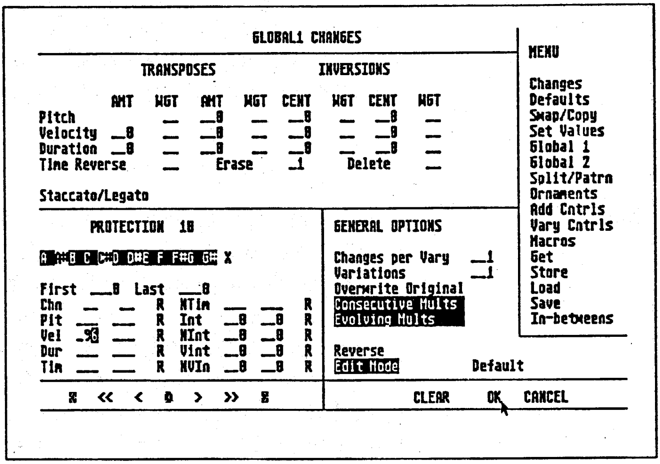
If you have a track with data on a number of channels, you can unmerge it into separate single-channel tracks by making several copies of the track, protecting the channel you want to keep in a given track, and using the **Erase** parameter on the **Global 1** screen to get rid of everything else. Single channels can also be split off by protecting all other channels and using the **SPLIT** button on the **Split/Pattern** screen.

Pit

Enter the lower and upper limits of the pitch range that you would like protected. Use note names and octave numbers (such as C3 - F#4). If you enter a pitch in one data field but not the other, the other will be set to its extreme limit.

Vel, Dur, & Tim

Data entered in these fields will have exactly the same effect with respect to velocity, duration, and time values. That is, events in a sequence whose values for any of these parameters fall within the protected range will not be varied. Protecting time or duration values might be an easy way to edit or vary certain portions of a sequence in a way dependent on the rhythm.



```
                           GLOBAL1 CHANGES
                TRANSPOSES              INVERSIONS
              AMT   WGT   AMT   WGT   CENT   WGT   CENT   WGT
Pitch               __    __0   __    __0    __    __0    __
Velocity      __0   __    __0   __    __0    __    __0    __
Duration      __0   __    __0   __    __0    __    __0    __
Time Reverse        __     Erase      _1      Delete      __

Staccato/Legato

        PROTECTION  10                 GENERAL OPTIONS

A A#B C C#D D#E F F#G G# X             Changes per Vary    _1
                                       Variations          _1
First ___0  Last ___0                  Overwrite Original
Chn  __   __   R  NTim ___  ___  R     Consecutive Mults
Pit  ___  ___  R  Int  __0  __0  R     Evolving Mults
Vel  _96  ___  R  NInt __0  __0  R
Dur  ___  ___  R  Vint __0  __0  R     Reverse
Tim  ___  ___  R  NVIn __0  __0  R     Edit Mode            Default

    %  <<  <  0  >  >>  %                     CLEAR    OK  CANCEL
```

PVG preset to erase all A#'s and C#'s with velocity less than 96.

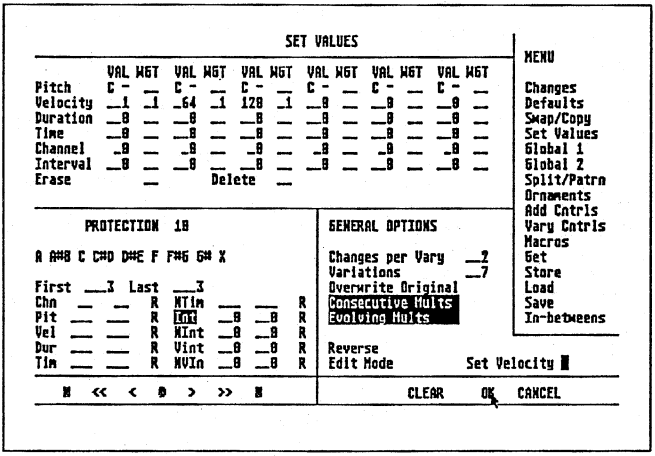
Int & VInt

The range of values entered in these fields will protect a note when its interval or velocity interval (velocity shift) with respect to the preceding note falls within the selected range. For example, protecting an **Int** range from 6 to 8 will prevent the program from making any changes in notes that are an augmented fourth, perfect fifth, or minor sixth above the preceding note. Falling intervals must be protected by negative numbers. Data entered under **Int** and **VInt** will have no effect unless you click on the **Int** or **VInt** abbreviation itself to switch it on.

NTim, NInt, & NVIn

These parameters operate exactly like **Int** and **VInt**, except that the program determines whether or not a note is protected by checking its relationship with the following note. Values entered under **NTim** will protect an event when the time of the next event (**NTim** means "next time") falls within that range. If you have improvised a number of quick phrases with long pauses between them, you might use **NTim** to protect the last note in each phrase, since the following note would have a large time value.

NInt works exactly the same way with reference to the interval relationship between the note about to be varied (or protected) and the following note. **NVIn** checks the velocity interval between the current note and the following note. **NInt** and **NVIn** must be switched on by clicking on the abbreviation itself.



```
                                 SET VALUES
                                                                    MENU
            VAL WGT  VAL WGT  VAL WGT  VAL WGT  VAL WGT  VAL WGT
Pitch       C -  __  C -  __  C -  __  C -  __  C -  __  C -  __   Changes
Velocity    __1  _1  _64  _1  128  _1  __0  __  __0  __  __0  __   Defaults
Duration    __0  __  __0  __  __0  __  __0  __  __0  __  __0  __   Swap/Copy
Time        __0  __  __0  __  __0  __  __0  __  __0  __  __0  __   Set Values
Channel      _0  __   _0  __   _0  __   _0  __   _0  __   _0  __   Global 1
Interval    __0  __  __0  __  __0  __  __0  __  __0  __  __0  __   Global 2
Erase            __      Delete    __                              Split/Patrn
                                                                   Ornaments
                                                                   Add Cntrls
       PROTECTION  10                 GENERAL OPTIONS              Vary Cntrls
                                                                   Macros
A A#B C C#D D#E F F#G G# X            Changes per Vary   __2       Get
                                      Variations         __7       Store
First ___3  Last ___3                 Overwrite Original           Load
Chn  __   __   R  NTim ___  ___  R    Consecutive Mults            Save
Pit  ___  ___  R  Int  __0  __0  R    Evolving Mults               In-betweens
Vel  ___  ___  R  NInt __0  __0  R
Dur  ___  ___  R  Vint __0  __0  R    Reverse
Tim  ___  ___  R  NVIn __0  __0  R    Edit Mode          Set Velocity

  ■   <<   <   ●   >   >>   ■                  CLEAR    OK   CANCEL
```

Preset to randomly set velocities of notes with rising intervals, i.e. notes with a pitch greater than that of the preceeding note. The first and last three notes of the sequence are not changed. 7 evolving variations are created, with two changes per variation.

This page intentionally left blank.

Chapter 5

Restrictions

Protection is a way of restricting the source material that the PVG may use in its operations. Restrictions, on the other hand, are a way of controlling the results of the operations. The **RESTRICTIONS** sub-screen is found at the lower left when the **Changes** screen is selected. Restrictions apply only to changes made from the **Changes** screen. (Protection also applies to **Changes**, though the **PROTECTION** sub-screen doesn't appear on this screen.)

Note that restrictions do not apply to the results of certain operations. For example, if you restrict the program to the notes of the C major scale and then enter a positive weight under **Interval** in the **Changes** section, black-key notes may very well appear in the output sequence. This is because **Interval** transposes everything in the source sequence that falls beyond the interval that is varied. The program does not check to make sure that all subsequent notes will fall on white keys as a result of the operation.

Scale Positions

When you highlight a scale position (by clicking on it) in the row below the word **RESTRICTIONS**, the program will not generate the notes you have highlighted. (To clear this row, simply click on the **X** at the right end.) If you have switched off **Allow New Scale Positions** in the **Defaults** box, then each time you enter the PVG, it will analyze the current sequence and preset this row accordingly. That is, if your sequence contains no F#s, then F# will be highlighted to prevent any F#s from being generated. This is an easy way of staying in the same key while creating a variation. (To change the **Allow New Scale Positions** switch, click on **Defaults** in the menu.)

Pitch Limits

This parameter allows you to set boundaries outside of which the PVG will not create notes. This is quite useful when lots of sizable pitch shifts are being generated, as the program can produce a surprising number of notes in the very top and bottom octaves of the MIDI note range, where most synthesizers don't sound their best.

Enter the values for **Pitch Limits** using note-name letters and octave numbers like those seen on the KCS edit screen. The specific notes entered are the upper and lower boundaries of the pitches that will be allowed; notes beyond those entered will not be allowed. If you enter only one value under **Pitch Limits**, the other value will be set to the extreme end of the range.

As with the individual scale position restrictions, **Pitch Limits** will not prevent notes outside the selected range from being generated if certain options such as interval variation are used. The new interval itself will always be inside the pitch limits, but the remainder of the sequence may easily wander too high or too low. (If this causes a musical problem, simply transpose the offending range of notes up or down an octave.)

Velocity Limits

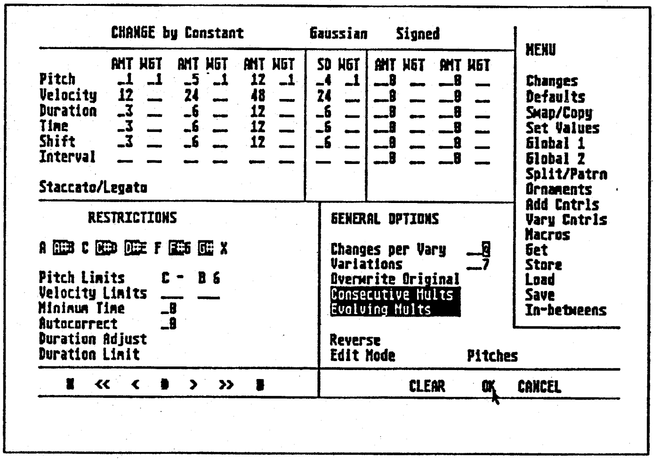
Use this item when varying velocity to prevent velocity values from getting too low or too high. As with **Pitch Limits**, entering only one value will cause the other to default, when the **Change** operation takes place, to the extreme end of the allowed data range.

Minimum Time

This parameter is used in conjunction with **Time** and **Shift** (compensated time) changes to prevent the PVG from jamming notes too close together. For example, if you are starting with a single monophonic line of even eighth-notes (time values of 12) and varying the **Time** and **Shift** parameters by 6, setting **Minimum Time** to 6 will prevent the program from generating any two-note unisons.

Autocorrect

The purpose of **Autocorrect** is just what you'd expect: It causes the new time values generated by the program to take on metrically even values. Another way of getting a similar result would be to set only metrically even values in the **Time** and **Shift** parameters. The difference between these two approaches becomes apparent when you are varying a sequence that isn't auto-corrected to begin with. In this situation, metrically even values in the **Time** and **Shift** parameters will continue to generate odd time values, depending on what the original times of the notes may be. That is, shifting a time value of 17 (odd) by 6 (a sixteenth note) will generate an odd value of 11 or 23. When an **Autocorrect** value is set under **RESTRICTIONS**, the program will generate only new time values that are multiples of the **Autocorrect** value (changing 17 to 12 or 18).



	CHANGE by Constant						Gaussian		Signed			
	AMT	WGT	AMT	WGT	AMT	WGT	SD	WGT	AMT	WGT	AMT	WGT
Pitch	_1	_1	_5	_1	12	_1	_4	_1	__8	__	__8	__
Velocity	12	__	24	__	48	__	24	__	__8	__	__8	__
Duration	_3	__	_6	__	12	__	_6	__	__8	__	__8	__
Time	_3	__	_6	__	12	__	_6	__	__8	__	__8	__
Shift	_3	__	_6	__	12	__	_6	__	__8	__	__8	__
Interval	__	__	__	__	__	__	__	__	__8	__	__8	__

Staccato/Legato

RESTRICTIONS

A A#B C C#D D#E F F#G G# X

Pitch Limits C - B 6
Velocity Limits ___ ___
Minimum Time _8
Autocorrect _8
Duration Adjust
Duration Limit

GENERAL OPTIONS

Changes per Vary __8
Variations __7
Overwrite Original
Consecutive Mults
Evolving Mults

Reverse
Edit Mode Pitches

MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
<< < > >>
CLEAR
OK
CANCEL

Preset to perform various random pitch changes. No sharps or pitches higher than B6 will be produced.

Duration Adjust & Duration Limit

When the **Duration Adjust** switch is highlighted, the program will prevent notes from overlapping by changing the duration of the note preceding any note that is shifted backward in time, in such a way that the duration of the preceding note will never be greater than the time value that separates them. When a note is shifted forward in time, creating a greater gap between the two, the duration is not changed. Note also that the duration of the shifted note itself is never changed; when shifted forward, it may now be overlapping a subsequent note.

Duration Limit has nothing to do with **Time** or **Shift** changes. It is a switch that prevents a duration change if that change would cause the note to overlap the next note. (The lower limit of duration changes is always 1; the program will not create notes of 0 duration.)

Chapter 6

Defaults

When you click on **Defaults** in the menu, a window will open in the center of the screen. The settings in this window cause the program to analyze the sequence being varied and set certain parameters in the **RESTRICTIONS** section accordingly. The values in the **Defaults** box are stored with each preset.

Limit Pitch Range

If a number is entered in this space, the program will find the highest and lowest notes in the sequence, and set the **Pitch Limits** in the **RESTRICTIONS** section to these outer limits plus the number of half-steps specified. For example, if the only pitch a sequence contains is C4, setting **Limit Pitch Range** to 6 would cause the **Pitch Limits** parameters in the **RESTRICTIONS** section to default to F#3 and F#4. (These values can later be edited by hand, of course.)

Limit Velocity Range

This parameter works just like **Limit Pitch Range**, but with reference to the velocity values in the sequence.

SD Default Factor

When this parameter is set, the program analyzes the source sequence and sets the values in the **SD** (standard deviation) column on the **Changes** screen to .1 times the **SD Default Factor** times the mean variation from one note to the next in the source sequence. **SD** values will thus be larger for sequences (and data types within those sequences) which have larger jumps from note to note than for sequences with small changes. (**Note:** Large values for the **SD Default Factor** can combine with large values for mean deviation to produce wraparound effects - settings in the **SD** column that don't correspond in any linear way to the explanation given above. If you don't like these, you can always edit them manually.)

Allow New Scale Positions

When this switch is switched off (not highlighted), the program will analyze the source sequence and highlight under **RESTRICTIONS** any scale steps that it doesn't find. This is a handy short-cut when you want all pitch changes to stay in the same key you started in.

Chapter 7

Changes

In the last three chapters, we've been dealing with various ways to control and limit the operations of the PVG. In this and the next few sections, we'll be looking at exactly how specific operations of PVG presets are set up.

When you first enter the PVG, you'll see the **Changes** screen across the top of the screen. The words **CHANGE by Constant** should appear at the upper left. If they don't, click on **Changes** in the menu bar.

AMT & WGT

You'll see a number of rows and columns of data fields in which parameter values can be entered. Several of the columns are headed **AMT** (amount) or **WGT** (weight). Understanding amounts and weights is crucial to understanding how the PVG works.

In order for the PVG to make a variation, you must tell it what sort of variation you would like. If you want to make several changes in the source sequence as part of a single operation, you must tell it not only which changes you would like made, but also how many you want in all, and how likely you want any individual change to be. You might want a lot of changes in velocity, for example, but only a few changes in duration and no changes in pitch.

Think of the total number of **Changes per Vary** (however large or small it may be) as a pie. When you enter data at any position in the weight columns, you are slicing up the pie. You are weighting the likelihood that any individual randomizing operation will be of the type you have chosen. In our example above, you might enter a weight of 10 in one of the **WGT** columns in the **Velocity** row, a weight of 2 somewhere in the **Duration** row, and no weights anywhere else. Your pie is now sliced conceptually into 12 equal pieces: 10 of them are velocity pieces and 2 are duration pieces. If you later change the velocity weight to 5 without changing the duration weight, the pie will only be sliced into 7 pieces. Duration will still have only 2 pieces of the pie, but each piece will be larger than it was before, because we're still dealing with a whole pie. In other words, reducing the weight under velocity will have the effect of increasing the probability that any given change will be a change in duration. Conversely, if you keep the velocity weight at 10 and the duration weight at 2 and add a pitch weight of 50, the likelihood that any individual change will be a duration change will now be very small (2 chances in 62).

Though it isn't immediately obvious on the screen, most of the **WGT** parameters are paired with **AMT** parameters in the columns immediately to their left. The **WGT** value determines the likelihood that a change will be made by that particular amount. For example, if you enter a **WGT** of 1 next to a pitch change of 1, you are creating some chance that a pitch-change of a half-step will occur in some note during the course of the variation.

The **AMT** columns default to some values that you may find useful. If not, you can overwrite them with values of your own. Both amounts and weights can be freely edited.

The **WGT** of some operations (such as **Delete** and **Erase**) does not appear in conjunction with any **AMT** parameter, because these operations are absolute. You can't erase some amount of a note! Other weights are paired with different parameters, but they operate in the same way, creating a positive probability (or certainty, depending on the nature of the parameter) that this particular change will occur. No change will be made in a data type for which no positive weight has been set.

If a 0 is set under **Changes per Vary** in the **GENERAL OPTIONS** section, the PVG will make one change for each weight unit. (The number of changes might appear to be slightly smaller, because some events might be varied more than once.) If **Changes per Vary** is not 0, there is no practical difference between giving five different parameters a weight of 10 and giving them each a weight of 1, because all the weights are equal. When **Changes per Vary** is 0, however, there is an enormous difference.

Constant, Gaussian, & Signed Changes

The **Changes** screen contains three sections, each of which has its own type of effect on the musical material. These are labelled at the top of the screen. You'll see six pairs of columns, though, rather than three. This is for convenience, and in order to make more complex variations possible. By entering positive weights next to three different amounts in one row, you can cause the program to generate changes of these three sizes in a single pass.

With **Constant** changes, you select the size of the change (and give the size a weight), but the program chooses the direction of the change. In the case of pitch changes, the pitch of a given note might be either raised or lowered. In the case of time changes, the note might get closer to the preceding note, or it might move further away.

If you'd like more control, you should use **Signed** changes. These two **AMT/WGT** columns at the right of the screen let you enter either a positive or negative value under **AMT**. (For negative values, type a minus (-) sign before the number. For positive values, simply type the number as usual.) Signed changes can be very useful if you want your material to start at one pitch or velocity level and move gradually, through a number of consecutive and evolving mults, to a higher or lower level.

Gaussian changes work a bit differently. Here the **AMT** column is replaced by a column headed **SD**. This stands for standard deviation, which is a statistical measure of the likelihood that a given change will be of some specified size. The value entered under **SD** will be roughly the mean value of the changes generated by the program.

To hear how Gaussian changes work, try entering a sequence that consists entirely of a single repeated note. Select **Consecutive Mults** but not **Evolving Mults**, enter 10 or 12 under **Variations**, and choose 0 **Changes per Vary**. Now enter a **WGT** of 10 or so and an **SD** of 1 in the pitch column, and click on **OK**. The result will be a sequence with many changes of a half-step, some changes of a whole-step, and a few larger changes. Next, go back to your source sequence again, reenter the PVG, and enter an **SD** of 5 or 6. This time, the average size of the pitch changes in the variations will be somewhat larger, but some changes of a half-step or whole-step will still appear.

The size of the standard deviation can also be controlled by the amount of variability in the source sequence itself. Larger amounts of variability (such as wide scale leaps) can cause a larger **SD** value to be calculated, resulting in wider changes in the variation. This is done by setting a value under **SD Default Factor** in the **Defaults** box (see Chapter 6).

Pitch, Velocity, & Duration

Random changes in these elements are made by entering one or more weights and amounts in the appropriate rows. The nature of pitch, velocity, and duration changes should be intuitively obvious. Note that the size and number of the changes may be affected by various settings under **PROTECTION** and **RESTRICTIONS.**

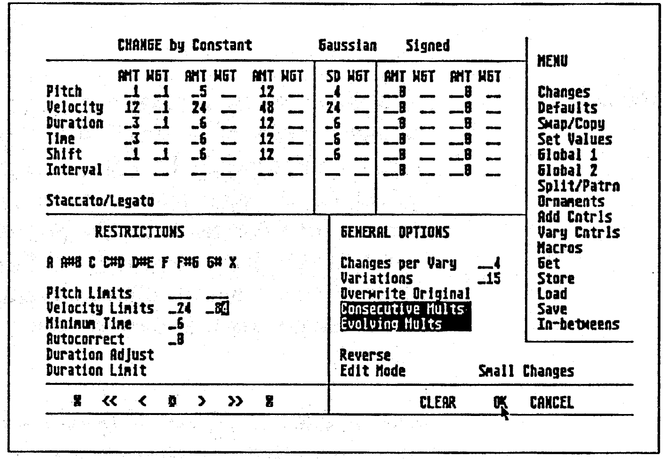
Time & Shift

These two parameters are related, in that they both operate on the time values of the MIDI notes in the source sequence. The difference is that when the time of a note is changed with **Shift**, a complementary change is made in the time value of the following note, in such a way that the overall length of the sequence is not changed. For example, if times are being shifted with an amount of 6 and all the notes in the original sequence are 12 clocks apart, then when the program shifts a note backward in time (subtracts 6 from its time value, giving it a new time value of 6) it will also add the same time value to the following note (12 + 6 = 18).

Using **Shift** is a great way to add syncopations to a sequence. You can use **Evolving Mults** and swapping to create a constantly evolving line, or simply generate repeated small variations on the same source material.

Time changes will change the overall length of the sequence. This can be useful for generating several closely related sequences that will gradually move out of sync with one another as they loop, because their lengths are no longer the same.

The PVG will not change the time of a note if the result of the operation would be to move it forward or backward past another note, changing the order of the notes. (If you want to change the order of the notes, use the **Swap/Copy** screen.) In this situation, the program counts the "change" as part of its total, but no change is actually made.



```
           CHANGE by Constant              Gaussian   Signed
                                                                      MENU
           AMT WGT  AMT WGT  AMT WGT    SD WGT   AMT WGT  AMT WGT
Pitch      _1  _1   _5  __   12  __     _4 __    __0 __   __0 __      Changes
Velocity   12  _1   24  __   48  __     24 __    __0 __   __0 __      Defaults
Duration   _3  _1   _6  __   12  __     _6 __    __0 __   __0 __      Swap/Copy
Time       _3  __   _6  __   12  __     _6 __    __0 __   __0 __      Set Values
Shift      _1  _1   _6  __   12  __     _6 __    __0 __   __0 __      Global 1
Interval   __  __   __  __   __  __     __ __    __0 __   __0 __      Global 2
                                                                      Split/Patrn
Staccato/Legato                                                       Ornaments
                                                                      Add Cntrls
      RESTRICTIONS                    GENERAL OPTIONS                 Vary Cntrls
                                                                      Macros
A A#B C C#D D#E F F#G G# X            Changes per Vary    __4         Get
                                      Variations          _15         Store
Pitch Limits     ___  ___             Overwrite Original              Load
Velocity Limits  _24  _84             Consecutive Mults               Save
Minimum Time     _6                   Evolving Mults                  In-betweens
Autocorrect      _0
Duration Adjust                       Reverse
Duration Limit                        Edit Mode            Small Changes

    0   <<   <   0   >   >>   0                 CLEAR     OK   CANCEL
```

Preset to produce various small random changes in pitch, velocity, and duration, and small shifts in time. The total length of the sequence will remain unchanged. No note will be moved less than 6 steps from the previous note, and velocities created will be in the range 24 - 84.

Staccato/ Legato

This switch at the bottom of the **Changes** and **Global 1** sections is used in conjunction with **Time** and **Shift** operations to preserve the relationships between the end of one note (that is, its duration) and the beginning of the following note (its time value). When a note moves backward in time, the note before it is shortened by the amount that it is moved, and its own duration is lengthened. When it moves forward in time, the note before it is lengthened, and the note itself is shortened.

This portion of the program operates by converting durations into a different form before the time changes are made, and converting them back afterward. As a result, some durations that are not actually involved in a particular change may be altered up or down by one unit. Also, using **Staccato/Legato** with a sequence that contains chords will tend to produce notes that have a duration of 1. (This can be fixed quickly using the **Chords** page of the Master Editor.)

Interval

The effect of an interval change is not to move the absolute pitch of a single note but to change the interval relationship between a single pair of notes. (Changing a single pitch changes two intervals - the one between that note and the preceding note, and the one between that note and the following note.) So an interval change has the effect of transposing everything in the sequence that follows the randomly selected interval.

As explained in Chapter 5, the program does not check to see that the transposed notes are still within the allowed note range, or that they still fall on allowed scale steps. This may seem at first to be a shortcoming of the program, but in fact it offers some interesting musical possibilities. If you don't like the non-key notes that are created by an interval shift, you can always use a global transposition along with protection of the notes that you do want, in order to put the offending notes back into the correct key.

Chapter 8

Swap/Copy

A good musical ear has a tremendous capacity for remembering patterns - that is, groups of notes that have specific relationships to one another. Because of this, introducing entirely new elements into the patterns, using the **Changes** section, may not always be the musical effect you're looking for. If you'd rather keep all of the existing elements intact, but rearrange them into a new order, the **Swap/Copy** section is the tool for the job.

Since the **Swap/Copy** operations only use the existing material in your source sequence, there is no need for a set of **RESTRICTIONS** parameters like those on the **Changes** screen in order to avoid creating new values.

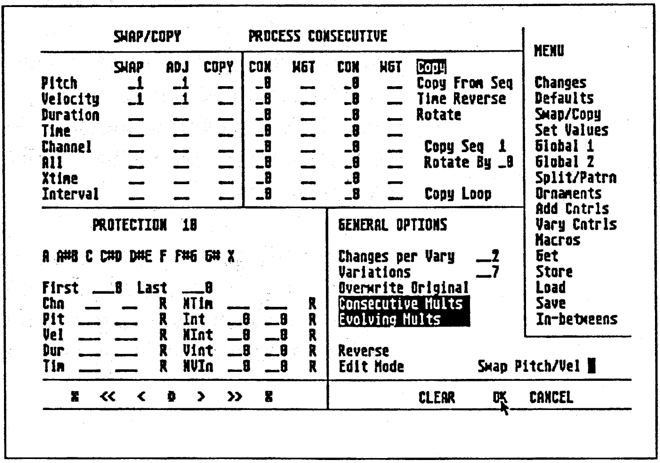
SWAP, ADJ, & COPY

Although they aren't labelled as such, all three of the columns on the left are **WGT** (weight) columns. That is, entering a number here creates a likelihood that the type of change selected will occur.

Entering a number under **Swap** causes the PVG to select two notes at random and copy the value of the selected type of data (pitch, duration, etc.) of each onto the other. Swapping a single data type a number of times is a good way to randomize the ordering of that data in the sequence without introducing any new values.

ADJ (Adjacent) is a special type of **Swap** operation that always uses two notes that are next to one another.

COPY is like half of a **Swap** operation. As before, two notes are chosen at random, but now the copying process is one-way rather than reciprocal. Using COPY a number of times will have the effect of reducing the amount of variety in the sequence. For example, if you start out with lots of different pitches and then copy pitches over and over, you could easily end up with a sequence that has only two or three repeating pitches in it. If you do this while simultaneously introducing an occasional new value from the Changes screen - well, who knows what you might get?



SWAP/COPY — PROCESS CONSECUTIVE

	SWAP	ADJ	COPY	CON	WGT	CON	WGT
Pitch	1	1		0		0	
Velocity	1	1		0		0	
Duration				0		0	
Time				0		0	
Channel				0		0	
All				0		0	
Xtime				0		0	
Interval				0		0	

Copy Seq 1
Rotate By 0

PROTECTION 10

A A#B C C#D D#E F F#G G# X

First 0 Last 0

Chn			R	XTim			R
Pit			R	Int	0	0	R
Vel			R	XInt	0	0	R
Dur			R	Vint	0	0	R
Tim			R	XVIn	0	0	R

GENERAL OPTIONS

Changes per Vary 2
Variations 7
Overwrite Original
Consecutive Mults
Evolving Mults

Reverse
Edit Mode
Swap Pitch/Vel

Copy
Copy From Seq
Time Reverse
Rotate
Copy Loop
MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
% << < 0 > >> %
CLEAR
OK
CANCEL

Preset to pick notes at random and swap their pitches or velocities with another note. The note following the selected note will be used half of the time.

Another trick you might like to try would be using **Reverse** in the **GENERAL OPTIONS** section along with the **Copy** operations, so that your evolving mults will be stored in the opposite order from the one in which they were generated. This could create a sequence that starts with only a few pitches (or velocities, durations, etc.) and then becomes gradually more complex in an interesting way. You can get a similar, though not identical, result by generating the variations without Reverse and then using the Time Reverse command (under the Transpose/Auto menu selection) in the KCS edit screen.

PROCESS CONSECUTIVE

Four different types of operations can be performed in this section, depending on which switch (**Copy, Copy From Seq, Time Reverse**, or **Rotate**) is highlighted at the right side. These switches are radio buttons: Only one of them can be active at a time, and it will govern all **PROCESS CONSECUTIVE** operations that are weighted.

Entering a number under **CON** tells the program how many consecutive notes you want it to perform the operation on. For example, entering a 3 on the pitch line under **CON** and giving this a weight will create a likelihood (or certainty, depending on what else the preset is or isn't doing) that the program will choose exactly three consecutive notes on which to perform the selected operation. Two **CON** columns are provided, so that you can perform identical operations on consecutive ranges of two different lengths at the same time.

Copy

Using the **Copy** switch under **PROCESS CONSECUTIVE** causes the program to select a consecutive group of notes (the number of notes being determined in the **CON** column) and copy the values of the selected data type to a target group of notes also selected at random. Doing this operation a number of times will reduce the overall variety in the sequence, but in a more interesting way than if you were copying only single notes. Short patterns will tend to replicate across the sequence, though not in any predictable fashion.

Copy From Seq

The **Copy From Seq** option works in conjunction with the **Copy Seq** parameter just below it. The program will go to the sequence whose number is entered under **Copy Seq**, choose a consecutive group of notes, and copy the selected data (pitch, duration, etc.) from these notes onto a consecutive group of notes in the sequence being varied. The result is that the variation will be a mixture of elements from two sequences, the sequence being varied and the sequence named under **Copy Seq.**

One interesting way to use this would be to create a "dummy sequence" of some specialized source material, such as a velocity accent pattern. The dummy sequence could be the same length as the number of notes entered under **CON**, in which case the same material would always be copied into the variation, or it could be longer, providing less predictable results. The difference between copying from a sequence with the PVG and doing a **Blend** operation with the Master Editor is that blending will affect the entire sequence, while the PVG will choose only a particular segment (or segments) of the sequence at random to copy data into.

Time Reverse

This option causes the values chosen (velocity, pitch, etc.) to be shuffled around within the consecutive region in such a way that they play back in opposite order.

Rotate

This option takes the values chosen and passes them down the line toward the end of the consecutive region, with the last few values wrapping around to the beginning of the region. The number of steps that each value will be advanced is determined by the **Rotate By** parameter.

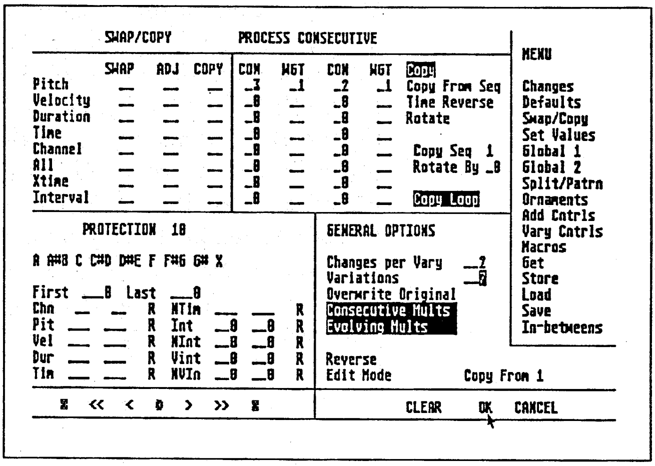
While **Rotate By** will not accept a negative number as an input, you can easily get the effect of rotating values backwards by entering a **Rotate By** value that is the desired number of steps less than the size of the consecutive region. For example, if the **CON** value is 9 and the **Rotate By** value is 7, the apparent result will be to rotate everything by -2.

Xtime

Most of the items listed at the left side of the **Swap/Copy** screen function just as you would expect them to, causing changes in the types of data indicated. One new item is found here, however. **Xtime** is an abbreviation for "except time." Entering a weight here will cause all of the elements of a note except its time to be affected by the operation. The result is that the rhythm of the variation will be the same as that of the original sequence, while other elements will have been shifted around. This could be particularly useful for creating variations to play along with composed music - even four-bar riffs in 4/4 time - because it will preserve the rhythmic structure while shifting the other elements.

Copy Loop

The **Copy Loop** switch in the lower right corner of the **Swap/Copy** screen works in conjunction with the **PROCESS CONSECUTIVE** options. When **Copy Loop** is activated, the program looks at the source sequence as a closed loop, and has the opportunity to define a region containing some notes at the end followed by some at the beginning.



```
           SWAP/COPY          PROCESS CONSECUTIVE
                                                                MENU
           SWAP  ADJ  COPY  CON  WGT  CON  WGT  Copy
Pitch       _     _    _    _3   _1   _2   _1   Copy From Seq   Changes
Velocity    _     _    _    _0   _    _0   _    Time Reverse    Defaults
Duration    _     _    _    _0   _    _0   _    Rotate          Swap/Copy
Time        _     _    _    _0   _    _0   _                    Set Values
Channel     _     _    _    _0   _    _0   _     Copy Seq  1    Global 1
All         _     _    _    _0   _    _0   _     Rotate By _0   Global 2
Xtime       _     _    _    _0   _    _0   _                    Split/Patrn
Interval    _     _    _    _0   _    _0   _     Copy Loop      Ornaments
                                                                Add Cntrls
     PROTECTION  10                  GENERAL OPTIONS            Vary Cntrls
                                                                Macros
A A#B C C#D D#E F F#G G# X           Changes per Vary    __2    Get
                                     Variations          __0    Store
First ___0  Last ___0                Overwrite Original         Load
Chn  _   _   R  NTim  ___  ___  R    Consecutive Mults          Save
Pit  __  __  R  Int   __0  __0  R    Evolving Mults             In-betweens
Vel  __  __  R  NInt  __0  __0  R
Dur  __  __  R  Vint  __0  __0  R    Reverse
Tim  __  __  R  NVIn  __0  __0  R    Edit Mode         Copy From 1

   ▣  <<  <  ▣  >  >>  ▣                     CLEAR    OK   CANCEL
```

Preset to copy 2 or 3 consecutive pitches selected at random from sequence 1 into the current sequence. Sequence 1 is treated as a loop.

Chapter 9

Set Values

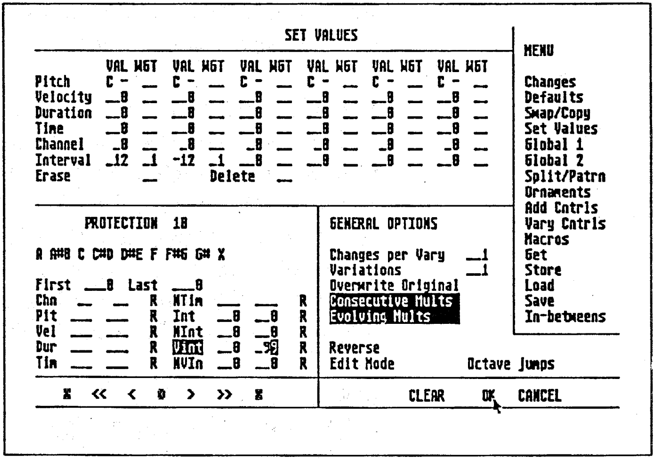
Operations that are selected in the **Set Values** section cause the program to select a single note at random and set it to the value specified. This is different from **Changes**, in that the result of a **Change** operation depends partly on the starting value of the note. For example, a change of velocity with a value of 24 would cause a note that started with a velocity of 60 to end up with a velocity of either 84 or 36. Entering a velocity of 24 on the **Set Values** screen will cause the randomly selected note to end up with a velocity of 24, no matter what velocity it started with.

Weights and amounts are set as in the **Changes** section. Six columns of changes with independent weights are available for each type of data, allowing you to make a number of set changes in a single variation. If **Changes per Vary** is set to 0, all of the set values you specify will be introduced into the variation. Otherwise, the program will select some of your set values at random each time it makes a variation.

Notes are entered in the pitch change **VAL** (value) columns as letter names (followed by a sharp sign, if needed) and an octave number.

Interval

Interval values may be either positive or negative. Unlike the **Interval** parameter in the **Changes** section, interval shifts here apply only to a single note; in other words, they operate much like pitch changes in the **Changes** section. The difference is that a pitch change generated in the **Changes** section causes a note to move up or down a specified amount with reference to its own starting pitch, while an interval shift in the **Set Values** section causes a pitch to move up or down into a specified relationship with the previous pitch.



SET VALUES

	VAL	WGT	VAL	WGT	VAL	WGT	VAL	WGT	VAL	WGT	VAL	WGT
Pitch	C -		C -		C -		C -		C -		C -	
Velocity	0		0		0		0		0		0	
Duration	0		0		0		0		0		0	
Time	0		0		0		0		0		0	
Channel	0		0		0		0		0		0	
Interval	12	1	-12	1	0		0		0		0	

Erase ___

Delete ___

PROTECTION 10

A A#B C C#D D#E F F#G G# X

First 0 Last 0

Chn			R	WTim			R
Pit			R	Int	0	0	R
Vel			R	WInt	0	0	R
Dur			R	VInt	0	99	R
Tim			R	WVIn	0	0	R

GENERAL OPTIONS

Changes per Vary 1

Variations 1

Overwrite Original
Consecutive Mults
Evolving Mults
Reverse
Edit Mode
Octave Jumps
MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
« < 0 > »
CLEAR
OK
CANCEL

Preset to randomly set notes to be an octave above or below the previous note. Only notes with velocities less than that of the previous note are affected.

Delete & Erase

Two new options are found in the **Set Values** section. Both **Delete** and **Erase** cause randomly selected notes to disappear from the variation. The difference between them is exactly the same as the difference between the **Delete** and **Erase** edit commands in the rest of the KCS: Deleting a note removes its time value from the sequence, causing the sequence to get shorter, while erasing an event causes its time value to be added to the immediately following event, so that the sequence stays the same length.

The **Erase** feature can be used to create wonderful syncopated bass lines. First, compose a sequence containing a bar or two of continuous sixteenth notes in whatever melodic pattern you like. Then erase some of them from the **Set Values** screen. Set the **GENERAL OPTIONS** to **Consecutive Mults** but not to **Evolving Mults**. (**Evolving Mults** with either **Delete** or **Erase** will quickly cause the entire sequence to disappear from the variations!) If your original sequence contains 32 sixteenth notes (two bars of 4/4), set **Changes per Vary** to 0 and give **Erase** a weight of 8 or 10. To add a bit more spice, you might also enter a signed velocity change of 12 on the **Changes** screen with a weight of 3 or 4, so as to cause occasional accents.

This page intentionally left blank.

Chapter 10

Global Changes

Two different screens allow you to make changes to all of the unprotected notes of a sequence. Unlike changes to single notes, global changes do not necessarily involve randomness. If a global change is the only change with a weight, or if a group of global changes have weights and the **Changes per Vary** parameter is set to 0, then all unprotected notes will be changed in a deterministic manner.

Global changes can be used to generate variations; one way to do this is to set a **PROTECTION** value of 5 and then protect some of the scale steps. Global changes can also be used to make complex edits. For example, protecting all of the scale steps belonging to a particular key and then transposing by a half-step (using a global pitch change) would bring any non-key notes up or down into the desired key relationship. presets containing only global changes can also be used as part of macros to perform quite complex edits on sequences, or to create macro variations. (See Chapter 17 for more on macros.)

To use a global preset to edit a sequence, click on **Edit Mode** under **GENERAL OPTIONS**. (This is equivalent to specifying 1 **Variations**, 0 **Changes per Vary**, and **Overwrite Original.**) Select the desired operations by giving them weights, and then protect the notes that you don't want to edit (using a **PROTECTION** factor of 10).

If your keyboard playing is sloppy, you can use global editing to eliminate any double-struck notes in a single-line sequence. Simply protect all notes with times larger than 2, and then select **Erase** in the **Global 1** screen. If you want to emphasize certain notes in the scale, protect the other notes and then transpose the velocity up, or set it to a high value. If you would like to double some of the notes in a line on a second MIDI voice module, copy the sequence, protect the notes you want to double, and then use global **Erase** (in conjunction with a global channel change, if desired).

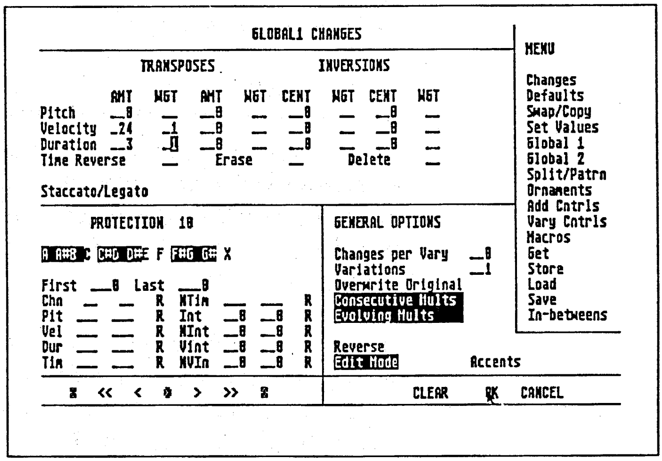
Global 1

The options on this screen are used for making the indicated type of change, either a transposition or an inversion in pitch, velocity, or duration, or time reversal or **Delete** or **Erase** operation.

TRANSPOSES & INVERSIONS

These two areas of the **Global 1** screen perform similarly to the equivalent functions in the KCS. (While two columns of amounts and weights are provided for each type of change, only one column will be needed if **Changes per Vary** is set to 0.)

Under **INVERSIONS**, a **CENT** (center) value is used to indicate the pivot point across which all values are to be mirrored. For example, a center value of 60 for **Velocity** would cause a note velocity of 59 to be changed to 61, a note velocity of 70 to be changed to 50, and so on. Center values for pitch must be entered as MIDI note numbers. (C4, which is Middle C on a standard 5-octave MIDI keyboard, is note number 60.)



```
                         GLOBAL1 CHANGES

              TRANSPOSES                  INVERSIONS

            AMT   WGT   AMT   WGT  CENT   WGT  CENT   WGT
Pitch       __0   __    __0   __   __0    __   __0    __
Velocity    _24   _1    __0   __   __0    __   __0    __
Duration    __3   _▯    __0   __   __0    __   __0    __
Time Reverse      __       Erase   __       Delete    __

Staccato/Legato

      PROTECTION  10                 GENERAL OPTIONS

A A#B C C#D D#E F F#G G# X           Changes per Vary   __0
                                     Variations         __1
First  ___0  Last  ___0              Overwrite Original
Chn  __   __   R  NTim  ___  ___  R  Consecutive Mults
Pit  ___  ___  R  Int   __0  __0  R  Evolving Mults
Vel  ___  ___  R  NInt  __0  __0  R
Dur  ___  ___  R  VInt  __0  __0  R  Reverse
Tim  ___  ___  R  NVIn  __0  __0  R  Edit Mode        Accents
```

MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
<< < > >>
CLEAR
OK
CANCEL

Preset to increase the velocity and duration of all C's, E's, and F's.

Time Reverse

This parameter does exactly the same thing as the **Time Reverse** switch in the Transpose/Auto section of the KCS edit screen: It reverses the order of the events in the sequence. There are two reasons for duplicating the function in the PVG. First, when **Changes per Vary** is not set to 0, entering a weight under **Time Reverse** causes some probability that a reversal will take place. And second, the PVG **Time Reverse** operation can be used in conjunction with the **Staccato/Legato** switch, which will change the durations of all the notes in order to preserve their relation to the following note (which used to be the preceding note).

Delete & Erase

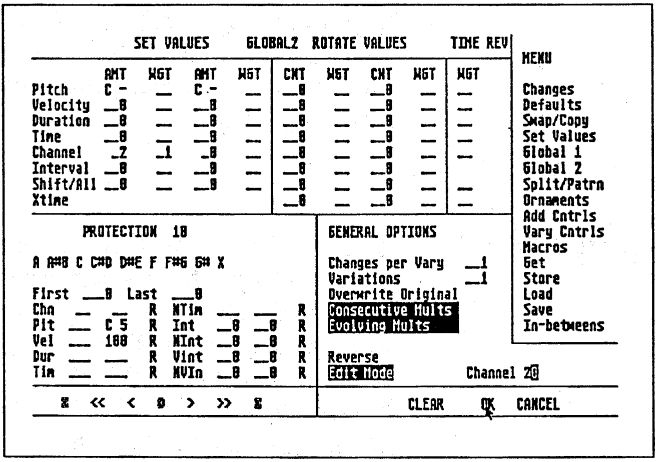
As usual, the numbers entered in these data fields are weights. Obviously, if you're using a single preset to create a variation, you will want to protect some notes when using **Delete** or **Erase**. If you don't, the variation will have no notes in it! Delete and Erase presets can be used creatively in macros to comb notes at selected bar or step positions out of a sequence.

Global 2

The functions available on the **Global 2** screen are more complex editing functions which have no close analog in the original KCS. All the data types that make up an ON event can be set to a fixed value, or parameters may be rotated or reversed within the sequence.

SET VALUES

Entering a weight in any of these fields will cause all unprotected notes to be set to the specified value. For example, you could set all occurrences of a particular pitch to a given MIDI channel by protecting all the other pitches. By performing such an operation several times, you could turn a one-channel sequence into a massively poly-timbral sequence.



```
              SET VALUES     GLOBAL2  ROTATE VALUES      TIME REV
                                                                  MENU
            AMT    WGT   AMT    WGT  CNT   WGT   CNT   WGT   WGT
Pitch       C -    __    C -    __   __0   __    __0   __    __   Changes
Velocity    __0    __    __0    __   __0   __    __0   __    __   Defaults
Duration    __0    __    __0    __   __0   __    __0   __    __   Swap/Copy
Time        __0    __    __0    __   __0   __    __0   __    __   Set Values
Channel      _2    _1     _0    __   __0   __    __0   __    __   Global 1
Interval    __0    __    __0    __   __0   __    __0   __         Global 2
Shift/All   __0    __    __0    __   __0   __    __0   __    __   Split/Patrn
Xtime                                __0   __    __0   __    __   Ornaments
                                                                  Add Cntrls
        PROTECTION  10               GENERAL OPTIONS              Vary Cntrls
                                                                  Macros
A A#B C C#D D#E F F#G G# X           Changes per Vary    __1      Get
                                     Variations          __1      Store
First  ___0  Last  ___0              Overwrite Original           Load
Chn   __   __   R  NTim  ___  ___  R Consecutive Mults            Save
Pit  ___  C 5   R  Int   __0  __0  R Evolving Mults               In-betweens
Vel  ___  100   R  NInt  __0  __0  R
Dur  ___  ___   R  Vint  __0  __0  R Reverse
Tim  ___  ___   R  NVIn  __0  __0  R Edit Mode             Channel 20

   %  <<  <  0  >  >>  %                     CLEAR    OK   CANCEL
```

Preset to put all notes with pitch above C5 and with velocities greater than 100 on MIDI channel 2.

ROTATE VALUES

A weight in either of the **ROTATE VALUES** columns will cause all of the values in the source sequence to be shifted downstream by the number of event-steps under **CNT** (count). As usual with **Rotate** operations, protection is applied to any notes specified under **First** and **Last**, but no protection is given to anything else.

If your source sequence contains a fair amount of variety, using several global **Rotate** operations at a time with different count values can be an interesting way to generate variations. Simply give each Rotate amount a weight of 1, and set **Changes per Vary** to 0. (Since these operations will be performed globally, you will get the same results every time you use the preset, so use **Evolving Mults** to get a constantly changing result.)

TIME REV

The right-hand column of weights in the **Global 2** screen is used to generate **Time Reverse** operations for the corresponding data type (pitch, duration, etc.). Creating a number of consecutive, evolving mults while time-reversing all of the values allows you to turn a short group of notes into a "back-and-forth loop." The major difference between this type of time reversal and others in the KCS is that here, the order of only a few of the parameters that make up a sequence are reversed, rather than reversing the order of the notes themselves.

Chapter 11

Split/Pattern

The **Split/Pattern** screen allows you to make the decision as to whether to process a note based on that note's relationship to up eight other notes in the sequence. It also allows you to perform splits based on those relationships, or to split out all the unprotected notes in a sequence.

The **Split/Pattern** screen might best be thought of as an extension of protection. It performs certain logical tests not on individual events, but on groups of events. Only notes that pass these tests will be varied. Pattern testing is done by the program at the same time as protection testing, so operations such as rotating, for which no protection testing is done, do not use pattern testing. Pattern testing works in conjunction with all presets that vary notes, but not with **Ornaments, Add Controllers**, or **Vary Controllers.**

Pattern testing works a bit differently than other areas of the PVG, so it may be a bit intimidating at first, but after you work with it for a while the concepts should become clear. The idea is that in the **Split/Pattern** screen the user specifies an interval pattern using the row of symbols. The pattern test looks for patterns which logically match the pattern you have specified. When the program is creating a variation and pattern testing is active (that is, when something other than **Off** has been selected in the row of buttons on the left side of the screen), only notes which meet the test criteria in the pattern test can be varied.

A pattern that is interrupted by any non-note event will not be identified by the program.

When the PVG is making a number of changes in a single variation, the pattern testing is done again before each new change. If a note has been changed, it may fail a logical test that it would have passed a moment before. This is true as well when the **Split** operation is performed: Notes that have been split are replaced by DE events, and will fail all logical tests.

There are four types of symbols in the **INTERVAL PATTERN SELECTION** area - end markers, note selector buttons, data fields, and arithmetic test symbols.

E (End)

The end markers appear as capital E's. These are radio buttons; only one can be highlighted at a time. An end marker is used to set the last note in the pattern that you are defining. For a two-note pattern, click on the **E** nearest the left end. This is above the second eighth-note symbol; there is no **E** above the first note symbol because the idea of a one-note pattern is meaningless. For a three-note pattern, click on the second **E** (the one above the third note), and so on. Any data in fields to the right of the **E** will be ignored.

Note Buttons

The note selector buttons, which appear as eighth-note icons on the screen, are used to indicate which of the eight (at most) notes in the pattern is the one you want to vary. Again, these are radio buttons, so only one of the notes in each selected pattern can be altered in the course of creating a variation (unless you select **Use Whole Pattern**). If you select the note button on the left end, each note that is to be varied will be tested to see if it can be the first note of the selected pattern; if you highlight the second note, each note will be tested to see if it can be the second note of the pattern, etc.

Interval Data Fields

The data fields are used to specify the size of the interval that is to be tested in identifying patterns. The fields operate like those in the pitch and interval areas of the **Changes** screen: Intervals are specified in number of half-steps, and negative numbers must be used to specify a descending interval. Data must be entered in the field between each pair of notes to the left of the currently active end marker. If you do not enter any data, the program will use a default value of 0 (a unison interval).

Arithmetic Tests

The arithmetic test symbols, which are the row of < = > icons below the interval pattern, are not radio buttons; they can be switched on or off in any combination. The symbols below each interval data field are used to tell the program what logical condition to test for with respect to that interval. If you select the equals sign for each interval (the simplest logical condition), then a pattern of notes in the sequence will be a match for your specified pattern only if all of the intervals are equal to the values you have specified. If you select the greater than sign, then all intervals larger than the value you have indicated will produce a match.

By clicking on more than one symbol at a time, you can specify the following logical criteria: Greater than, less than, equal to, less than or equal to (both the less than and the equal symbol are highlighted), greater than or equal to (the greater than and equal symbols together), not equal to (the greater than and less than symbols, but not the equal symbol), or any interval (all three highlighted). This last option appears a bit silly, but it could be useful, for example, to vary the third note before any interval of an octave, no matter what the intervening intervals might be.

Use if Match & Use if no Match

The **Use if Match** and **Use if no Match** buttons tell the program what to do when it finds a pattern that matches your criteria. If you select **Use if Match**, then notes will be varied only in the patterns you have specified. If you select **Use if no Match**, patterns that match your criteria will be protected, and notes in all other patterns will be varied.

Use Whole Pattern

This button overrides the setting of the note symbols in the pattern row. When **Use Whole Pattern** is selected, each note to be varied will be tested to see if it can be the first note of the selected pattern. The variation process you have specified in your Vary Notes preset will be applied not to the single note indicated but to all of the notes in the pattern as if they were a single note. (Since a pattern must consist of more than one note, the total number of notes varied by the preset will usually be several times larger than the number specified by the **Changes per Vary** parameter or the weight parameters.)

By specifying a pattern in which all of the arithmetic test icons are highlighted, you can specify that the same random change or **Set Value** operation will be applied to up to eight consecutive notes. Note that the **Use Whole Pattern** feature cannot be used with **Swap** and **Copy** operations.

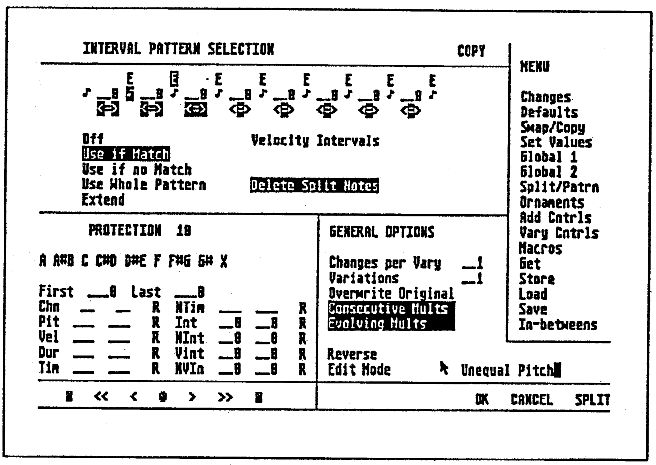
Extend

The **Extend** operation causes the program to test the notes of the pattern as before, but to continue to recognize as part of the pattern any additional notes that pass the test you specify under the last interval (the one at the right end of the pattern row). This allows you to process a longer pattern, if it consists of some regular shape such as rising or falling intervals, for as long as it lasts. If **Extend** is selected, the first note in the pattern will always be the one that is varied; the note button is overridden.

If any of these options other than **Off** is selected, all of the other screens that control Vary Notes presets will display the words **Using Preset** to the right of the **GENERAL OPTIONS** label.

Velocity Intervals

This switch is used to test for patterns in note velocity rather than pitch. As with pitch, the velocity intervals test is applied not to individual velocities but to relations between adjoining pairs of velocities.


INTERVAL PATTERN SELECTION

Off
Use if Match
Use if no Match
Use Whole Pattern
Extend

Velocity Intervals
Delete Split Notes

PROTECTION 18

A A#B C C#D D#E F F#G G# X

First ___8 Last ___8
Chn __ __ R NTim ___ ___ R
Pit ___ ___ R Int __8 __8 R
Vel ___ ___ R NInt __8 __8 R
Dur ___ ___ R Vint __8 __8 R
Tim ___ ___ R NVIn __8 __8 R

GENERAL OPTIONS

Changes per Vary __1
Variations __1
Overwrite Original
Consecutive Mults
Evolving Mults

Reverse
Edit Mode
Unequal Pitch

COPY
MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
<< < > >>
OK
CANCEL
SPLIT

Pattern selection screen to process notes only if their pitches are different from the pitches of the preceding AND following notes.

SPLIT & Delete Split Notes

In addition to the **OK** and **CANCEL** exit buttons found on every screen, the **Split/Pattern** screen has a **SPLIT** button. When you tell the PVG to perform a **SPLIT** operation, it checks all of its **PROTECTION** parameters, including the pattern matching if this is active, and copies all of the unprotected notes to a new sequence. If you switch on **Delete Split Notes**, the copied notes will also be deleted from the original sequence. Since the protection criteria are quite diverse, you can easily split material of a number of different types, such as notes with long or short duration or notes of a given pitch or pitches, and park them in a sequence by themselves for further editing.

This page intentionally left blank.

Chapter 12

In Betweens

Another basic operation that the PVG can perform is to create one or more sets of sequences (or consecutive mults) that are linear interpolations between any two existing sequences. The in-between generator does not use any of the settings on the main PVG screens. When you activate it by clicking on **In-betweens** at the bottom of the menu, its window will pop up. In-betweens can be created only in Open mode.

Note that there is no stipulation that the two sequences used for in-betweens must have the same number of notes in them. In fact, some of the most striking results of the in-between feature come from using a starting sequence that has only a single note and an ending sequence that consists of a whole phrase. (Autocorrecting the in-betweens is also a good idea, if you're interested in increasing the understandability of the output.) Each of the in-betweens will have some number of notes intermediate between the number of notes in the starting and ending sequences.

Sequence

Enter on this line the number of the sequence that you want to use as the end-point for the in-between process. (The sequence from which you entered the PVG will be the starting point.)

Number of Sets

This number determines how many interpolations the program will make between the starting sequence and the ending sequence. It is not a linear setting, however. The number of sequences created is equal to $2^n - 1$, where n is the value in this field. The range of n is 1 to 5. The number of sequences created (or the length of the combined sequence, if **Consecutive Mults** is selected) can get to be quite large when this setting is 4 or 5.

Autocorrect	This value works exactly like the **Auto-correct** value in the KCS edit menu; it restricts the time values of the generated sequence(s) to multiples of the number entered in this data field. Autocorrecting is especially useful when generating in-betweens for sequences that have different rhythms or different numbers of notes, as it will make the output sound more coherent.
Consecutive Mults	This switch works exactly like the one in the **GENERAL OPTIONS** section. It causes all of the in-betweens to be stored consecutively in the same output sequence.
Restrict Scale Positions	This switch is used for telling the In-Between generator what scale positions you would like it to use. If you click on **OK** to start generating in-betweens when the **Restrict Scale Positions** switch is highlighted, you will get a second window with a scale positions display like the one in **RESTRICTIONS and PROTECTION**. This defaults to the notes contained in the starting and ending sequences, but you may select any combination of scale steps that you prefer.

Chapter 13

Ornaments

Up to now, we've been looking at PVG operations that maintain a one-to-one correspondence between the notes of the starting sequence and the notes of the newly created sequence. The notes may be so jumbled or transformed as to be unrecognizable, and a number of variations may be strung together so that the total number of notes increases, but no new notes will be added to the material during the course of any single variation.

Adding new notes, however, is one of the basic ways that musicians vary their material. Doing an entire 32-bar improvised chorus is beyond the scope of the PVG (though some other computer software can do this). What the PVG can do is add new notes around existing ones, using Ornaments presets. In its most basic usage, the ornaments function can add grace notes or trills to randomly or non-randomly selected notes, much the way a human musician would. You'll find, however, that the ornamentation process can create passages of startling complexity and beauty.

Note that, unlike some of the ornaments performed by human musicians, PVG ornaments never alter the basic value of the notes being ornamented. A triple grace note that lies across the beat, for example, will not cause the main note to start later and have a shorter duration. If you want this effect, you have two choices: Either time-shift and shorten the main note manually on the edit screen, or use the PVG **Global** screens to edit all of the notes you're planning to ornament before performing the ornamentation. (When notes to be ornamented are being selected at random, only the first process can be used.)

The ten columns on the upper side of the **Ornaments** screen are used for defining up to ten ornamental notes, which will be added in relation to existing notes in the sequence. The meanings of the entries under **Pitch**, **Velocity**, and **Duration** will be affected by the settings of the switches immediately below, at the left center of the screen.

PROTECTION and **GENERAL OPTIONS** apply to the ornament process exactly the way they do to any other type of variation. When **Changes per Vary** is set to 0, the program will add ornaments to all unprotected notes. When it is set to some value greater than 0, unprotected notes will be chosen at random for ornamentation.

Offset & Delay

These two settings together determine the rhythmic shape of the ornament. **Offset** is the starting time of the ornament in relation to the note being ornamented, and can be either positive or negative. For a simple grace note, you might use an **Offset** of -6, which (at normal clock resolution) would cause the ornament to start a sixteenth note before the main note.

The numbers entered in the **Delay** row determine the starting times of the notes in the ornament in relation to the offset time. All delay times must be positive; that is, the notes in the ornament cannot start before the beginning of the ornament. A delay time of 0 will cause a note to start precisely at the offset time, while larger delay values will cause notes to start correspondingly later.

Note that this system does not force ornament notes to fall immediately before or during the sequence note that they are "ornamenting." They can be offset by relatively large amounts, resulting in less-than-obvious musical relationships. If a large negative offset would cause ornament notes to fall before the beginning of the sequence, they are not generated.

Note also that there is no requirement that the ornament notes be listed on the table in direct rhythmic order from left to right. Normally you will probably want to set the delay of the first note to 0 and use ascending delay amounts as you move to the right; this will make it easier to keep track of what is going on in your ornament. But if you should want to try out the notes of an ornament in some other order, you don't need to retype them from scratch: Simply enter new delay times. Entering identical delay values for several ornament notes will cause them to be played as a chord.

Pitch, Velocity, Duration, & Channel

Values entered in the columns on the **Pitch**, **Velocity**, and **Duration** rows can be either absolute or relative to the note being ornamented. When **Follow** is highlighted for that parameter type, they are relative; when **Fix** is highlighted, they are absolute.

For example, when **Follow** is switched on for **Pitch**, entering a 2 under **Pitch** will cause an ornament note to be generated a whole-step (two half-steps) above the note being ornamented. When **Fix** is switched on, however, a 2 will cause MIDI note 2 (D-1) to be generated. In the same way, a value of -6 under **Duration** will cause the ornament note to have a duration 6 clock units shorter than the note it is ornamenting.

Pitch, **Velocity**, and **Duration** values can be either positive or negative; but obviously, negative settings are more useful when **Follow** is selected. Fixed negative values would be meaningless, so the program sets any "negative" notes to the lowest permissible values (unless you select **Cut**, as described below).

The function of the **Channel** parameters should be obvious: Using these, you can assign each ornament note to its own MIDI channel. This might be useful for exotic effects, or for layering one or more of the notes in an ornament with a sound from a second module. When there is no entry under **Channel**, all ornament notes will default to the channel of the note that they are ornamenting.

Cut

When **Cut** is switched on to the right of the **Velocity** or **Duration Fix/Follow** switch, ornament notes that have negative velocities or durations will be eliminated. (When **Cut** is switched off, these notes will still appear in the output sequence, but will have a velocity or duration of 1.)

COPY

There may be times when you would like to use a bit of material that is already recorded in a sequence as ornamentation for this or some other sequence. You can load the ornament table automatically by clicking on **COPY**. The table will be loaded with the notes at the beginning of the current sequence, or the current range if one is lit up on the edit screen. The program will set up the ornament table so that the first ten notes after the first (that is, notes 2 through 11) would be added to the first if the ornamentation process were applied to it. The copy will stop before ten notes if it encounters something other than a note (such as a program change or controller message) in the sequence or range.

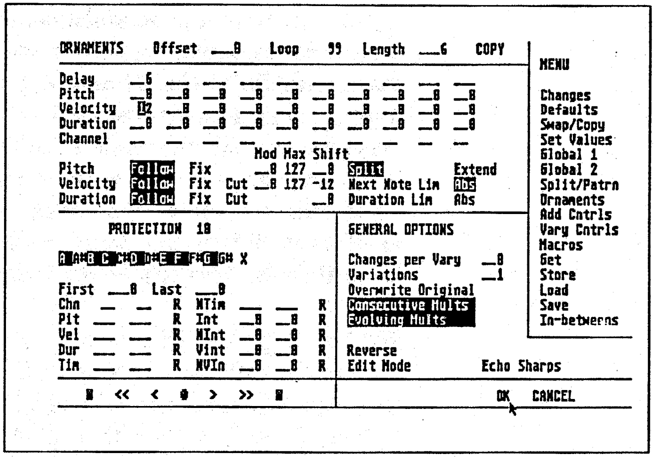
The program checks to see whether you have selected **Fix** or **Follow**, and loads the table accordingly.

Loop, Length, & Shift

There may be times when you want an ornament to consist of a small number of notes that are repeated several times. (For example, a trill consists of two notes repeated several times.) To do this, simply enter the note group once in the ornament table, and then enter the number of times you want this group to repeat under **Loop**. The **Loop** value must always be at least 1; if it were 0, no ornament would be generated.

The **Length** parameter is used to specify how far apart the iterations of the loop will be. (It does not actually determine the length of the ornament, which could be longer or shorter.) For example, if **Offset** is set to -24, **Loop** to 4, and **Length** to 6, the program will generate four identical ornaments for each note that it acts on, one starting 24 clocks before the note, one 18 clocks before, one 12 clocks before, and one 6 clocks before.

The values under **Shift** determine how each iteration of the loop will relate to the previous one. For example, if you want an ornament to start with low velocities and get louder each time it is added to the sequence, you should use a positive shift value in the **Velocity** row. Shifting the pitch will cause each successive ornament to be higher or lower than the one before; with large loop values, you can create some wonderful cascading effects using **Shift**.



```
ORNAMENTS   Offset __0   Loop  99  Length __6   COPY
Delay     _6  __  __  __  __  __  __  __  __  __
Pitch     _0  _0  _0  _0  _0  _0  _0  _0  _0  _0
Velocity  12  _0  _0  _0  _0  _0  _0  _0  _0  _0
Duration  _0  _0  _0  _0  _0  _0  _0  _0  _0  _0
Channel   _   _   _   _   _   _   _   _   _   _
                          Mod Max Shift
Pitch     Follow  Fix      _0 127 _0  Split        Extend
Velocity  Follow  Fix  Cut _0 127 -12 Next Note Lim Abs
Duration  Follow  Fix  Cut        _0  Duration Lim  Abs

      PROTECTION  10                GENERAL OPTIONS

A A#B C C#D D#E F F#G G# X          Changes per Vary  _0
                                    Variations        _1
First __0  Last __0                 Overwrite Original
Chn  _   _   R  NTim  __  __  R     Consecutive Mults
Pit  __  __  R  Int   _0  _0  R     Evolving Mults
Vel  __  __  R  NInt  _0  _0  R
Dur  __  __  R  VInt  _0  _0  R     Reverse
Tim  __  __  R  NVIn  _0  _0  R     Edit Mode         Echo Sharps
```

MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
<< < > >>
OK
CANCEL

Preset to echo all # notes every 6 time steps until the next note, or the end of the sequence, is reached. The first note will have a velocity 12 steps higher than the original, but each subsequent note will have its velocity reduced by 12. The echo notes will be stored in the next available sequence.

Mod & Max

There may be times when you want the number of loops in the ornament to depend on the material being ornamented. For example, you might want to attach long trills to high notes and shorter trills to low notes, or add longer ascending glissandi to notes with low velocities than to those with high velocities. In this situation, you will use the **Mod** and **Max** (modulation and maximum) parameters. **Mod** and **Max** apply only to the looping process, not to single ornaments.

The number of loops can never be greater than set by the **Loop** parameter. Modulation will have the effect of making the number of loops smaller. The value under **Max** is the point in the velocity or pitch range beyond which you want the modulation to have little or no effect. For example, entering a **Max** of 60 under pitch makes Middle C (MIDI note 60) the upper or lower limit of the modulation. When the **Mod** value is negative, notes above the **Max** will be modulated (that is, be ornamented with fewer iterations of the loop), while those below the **Max** will be relatively unaffected. When the **Mod** value is positive, notes lower than the **Max** will be modulated.

For more predictable results, we suggest that you experiment with various **Mod** and **Max** settings using a test sequence consisting of a single pair of notes, one high and one low in either pitch or velocity. Once you're clear what effect the settings have with two notes at the extremes of the range, fill in some additional notes in the middle and note how the curve operates.

Split

This flag, when highlighted, causes the ornaments to be stored on the next available sequence or track rather than on the original. One variation is created, and the **Variations**, **Overwrite Original**, and **Edit Mode** settings are ignored.

Extend

This flag allows new notes to be added that would fall after the end of the original sequence or track.

Next Note Lim, Duration Lim, & Abs

These flags are used to control in various ways whether or not the looping material will extend past the end of the note being ornamented. When **Next Note Lim** (limit) is switched on, no new loop will be started after the beginning of the note following the ornamented note, no matter how many loops the program is set to create. When **Duration Lim** is switched on, no new loop will be started after the end of the current note (that is, after its duration is complete).

The two **Abs** (absolute) flags are similar in function to the flags immediately to their left. While **Next Note Lim** prevents a new loop from being started after the beginning of the note following the ornamented note, it does not shut off the current loop, which may continue sounding for some time, depending on how many notes are in it. The adjacent **Abs** flag, on the other hand, will shut off the current iteration of a loop, at whatever point it may have reached, when the new note in the sequence is reached. The **Abs** flag next to the **Duration Lim** flag does the same thing with reference to the current note's duration - that is, it shuts off the current iteration of the loop whenever the note being ornamented ends. The duration of ornament notes is not affected by the shut-off, however.

This page intentionally left blank.

Chapter 14

Add Controllers

Add Controllers presets can be used to insert controller messages after randomly selected notes in a sequence, or after all unprotected notes. Adding controllers is much like adding ornaments, except that controller events are generated instead of notes. The controller messages can be any of the defined MIDI continuous controllers (CC events), after-touch (AT events), or pitch-bend (PB events). Up to 18 controller events can be added to each note, and both the amount and the timing can be modulated by the pitch or velocity of the note. Longer strings of controller data can be generated by looping. In addition, presets of this type can be used to add program change messages before notes.

Delay, Type, & Value

The controller events that are to be added are defined in the six rows of data fields in the upper part of the screen, after the words **Delay**, **Type**, and **Value**. These three parameters define a controller change, and there are 18 cells of three data fields each on the screen. If you like, you can consider this area as three long rows of data rather than as six shorter ones: The three fields placed in each vertical group work together, but there is no logical linkage between the upper three fields and the lower three in the same column.

The **Delay** parameter defines how many time steps after the note the controller event is to be inserted. Delay times are all relative to the note, not relative to previous delay times. To get, for example, a series of controllers separated by 1 step, enter delay values of 1, 2, 3, 4, and so on. A delay time of 0 will cause the event to fall on the same step as the note, but it will still be inserted in the file, and sent out over MIDI, after the note-on event. Unlike delay times in Ornaments presets, those in Add Controllers presets cannot be negative; the logic here is that before the note starts sounding, there is nothing to control.

Normally, you will probably want to set the delay times in ascending order; it makes the screen easier to understand. However, controller events can be entered in any order. The program will not generate new controller events following a non-note event that is already in the sequence, but it will generate the new controller events without reference to their order on the **Add Controllers** screen.

If the **Type** parameter is a positive integer, it corresponds to the MIDI controller number (0 - 127). A -1 is used to specify after-touch (AT) events and a -2 to specify pitch-bend (PB) events. Note that some of the continuous controllers are defined by the MIDI Specification in special ways; it's up to you and your synthesizer to determine the exact meaning of the controller events generated by the PVG.

For advanced applications, you may find it entertaining to use several different types of controllers in a single preset. Using the **Loop** parameter to generate cycles of pitch-bends, modulation wheel movements, and loudness changes (controller 7) on a single sustained note can produce some quite striking results. If you have a Yamaha DX series instrument, try programming some patches to respond to breath controller (controller 2).

The value parameter can be set anywhere in the range from -8192 to 8191. This is because MIDI pitch-bending is defined as a two-byte event (even though most synthesizers discard a number of the bits in the message). Controllers and after-touch have meaningful values from 0 to 127, but the PVG will not prevent you from assigning absurdly large or negative values in a preset; it just won't insert them into the sequence.

Pgm & Offset

These two data fields, at the top left of the screen, are used to add a program change (PG event) before a note or notes in the sequence. The number entered under **Offset** tells the program how many clock steps before the note you want the PG event to appear. The number entered must be positive, but the program will understand it as essentially negative - that is, pointing backwards in time.

The number of the program change cannot be varied; it will always be exactly what you specify. However, there are a couple of ways to introduce variability. When using an individual Add Controllers preset to create a variation, you could specify something other than 0 **Changes per Vary**. In this situation, the PVG will choose a note or notes at random before which to insert the program change. It's also possible to set up a whole group of Add Controllers presets to do nothing but generate PG events, and then select these at random from a macro.

Loop & Length

These parameters operate exactly like their counterparts in Ornaments presets. The value under **Loop** tells the program how many times to generate the series of controller events you have defined, and the value under **Length** tells it how far apart the beginnings of successive iterations of the series are to be. The **Length** value can be smaller than some of the **Delay** values for individual events; this will cause the iterations of the loop to be embedded among one another.

COPY

This button, at the upper right corner of the screen, is used to fill the data fields with controller events from the current sequence. The program expects to see one note (an ON event) followed by controller data. It will not begin filling the table unless it sees an initial note, and it stops filling the table when it encounters a second note.

Split

When highlighted, this button will cause the controller events that are generated to be stored on a track by themselves, rather than on a track that also contains the notes from the sequence being varied.

Next Note Lim, Extend, & Duration

These flags are used to tell the preset when to stop adding controllers to a sequence. This is especially useful when a loop is being used, but will work just as well with single sets of controllers.

When **Next Note Lim** (limit) is highlighted, the program will not generate any controller events after the beginning of the note following the note to which they are logically attached. When **Duration** is highlighted, new events will not be generated after the end of the current note as defined by its duration value. When **Extend** is highlighted, the program will continue to generate controller events, if required by the definitions in the preset, after the final DE event in the sequence. Otherwise, the DE event will be considered the boundary of the variation, and no events will be generated beyond it.

Modulation

The timing (delay parameter) and values of the controller events created by a preset can be modulated by the pitch or velocity of the note to which controllers are being added. The modulation section is in the left center of the screen, where the columns are labelled **Mod** and **Fix** and the rows **Pitch** and **Velocity**. The four data fields on the left modulate the values of the controller events, while the four to the right of the word **Delay** modulate the time delay of the events.

Except for the parameter that is being modulated, the two types of modulation work exactly the same way. Modulation works a bit differently in **Add Controllers** than it does in **Ornaments**, however: The number of iterations of the loop cannot be modulated, but modulation can be either positive or negative with respect to the starting values (those programmed in the top half of the screen).

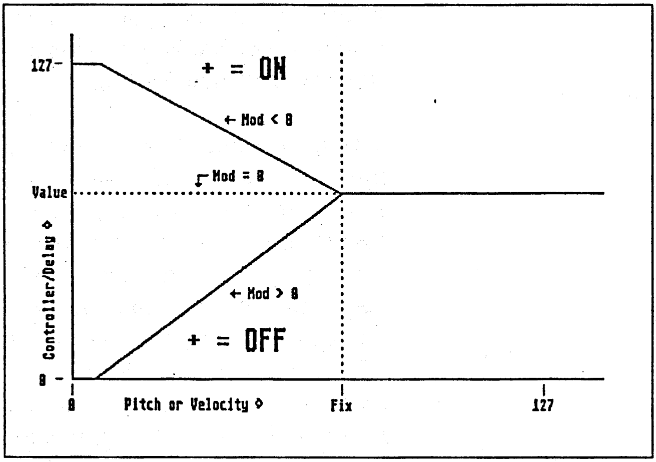
In fact, there are two ways for modulation to be positive and two ways for it to be negative, leading to four possible logical conditions. The value of the **Mod** parameter can range from -99 to 99, and in addition the plus-sign icon to the right of each pair of **Mod/Fix** parameters can be switched on or off. When the **Mod** value is positive, controller values will be higher for higher values of the modulating parameter (the pitch or velocity of the note to which controllers are being added). When the **Mod** value is negative, the reverse condition occurs: Higher pitches or velocities will lead to lower controller values.

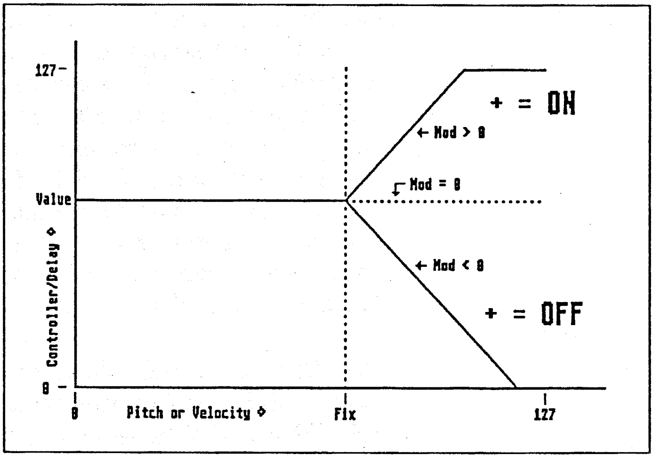
The value entered on the top half of the screen is an upper or lower limit. Modulating this value from a single modulator can increase it or decrease it, but cannot do both. (Modulating from both pitch and velocity at the same time, with opposite signs, will allow the basic value to be both increased and decreased during the course of a single variation.) When the plus sign is highlighted, modulation will increase the value above this limit; that is, it will be a lower limit. When the plus sign is switched off, the limit will be an upper limit, and modulation will decrease the values.

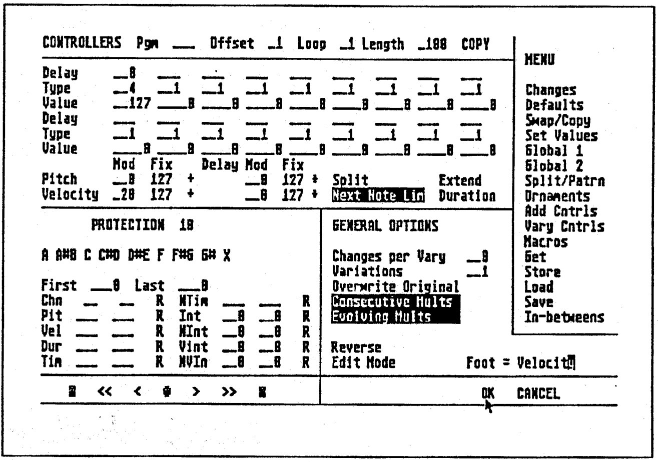
The **Fix** parameter links the limit value shown in the top half of the screen to a specific pitch or velocity value. For example, if a **Fix** value of 60 (corresponding to C4) is entered on the **Pitch** row, then the controller values, if modulated by pitch, will always be at the values entered in the top half of the screen when the controller events have been added to a C4. Whether they will be higher or lower than this with higher or lower pitches depends on the signs attached to the modulation, as explained above.

The effects of modulation in the PVG are complex, and you will probably need to experiment to get the feel of things. If **Value** is set to 127, with **Velocity Mod** equal to 20, **Fix** at 127, and + turned off, the controller values added will exactly equal the velocity values of the preceding notes. If **Fix** is equal to 0, **Value** is 32, and + is on, then a note with 0 velocity will yield a controller value of 32, and a note with velocity 127 will get twice that, or 64. The **Delay Mod** settings allow you to control the timing of a controller sequence with the pitch or velocity of a note.

The basic relationships between the **Mod, Fix, Value,** and **+** parameters are illustrated in the following figures:







```
CONTROLLERS  Pgm ___  Offset _1  Loop _1 Length _100  COPY

Delay     __0   ___   ___   ___   ___   ___   ___   ___   ___
Type      __4   __1   __1   __1   __1   __1   __1   __1   __1
Value     __127 ___0  ___0  ___0  ___0  ___0  ___0  ___0  ___0
Delay     ___   ___   ___   ___   ___   ___   ___   ___   ___
Type      __1   __1   __1   __1   __1   __1   __1   __1   __1
Value     ____0 ___0  ___0  ___0  ___0  ___0  ___0  ___0  ___0
          Mod   Fix     Delay Mod   Fix
Pitch     __0   127  +        __0   127 +  Split          Extend
Velocity  _20   127  +        __0   127 +  Next Note Lin  Duration

      PROTECTION  10               GENERAL OPTIONS

A A#B C C#D D#E F F#G G# X         Changes per Vary    __0
                                   Variations          __1
First ___0  Last ___0              Overwrite Original
Chn  __   __   R  NTim  ___  ___ R Consecutive Mults
Pit  ___  ___  R  Int   __0  __0 R Evolving Mults
Vel  ___  ___  R  NInt  __0  __0 R
Dur  ___  ___  R  Vint  __0  __0 R Reverse
Tim  ___  ___  R  NVIn  __0  __0 R Edit Mode        Foot = Velocity
```

MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
<<
<
>
>>
OK
CANCEL

Preset to add a foot pedal CC event (type 4) after each note, with the controller value equal to the notes' velocity.

This page intentionally left blank.

Chapter 15

Vary Controllers

Unlike Add Controllers presets, which introduce new controller data into a sequence, presets of the Vary Controllers type are used to make random or deterministic changes in controller data that is already present in a sequence. The two types of presets can, of course, be used in complementary ways in a macro. Either or both of two types of operations can be carried out in a Vary Controllers preset - **Global** operations, which affect all of the controller events within a sequence that are of the specified type, and **Single** controller operations, which operate on individual controller events.

To specify which controller events you would like to vary with the preset, enter a MIDI channel and a controller number (0 through 127, -1 for after-touch, or -2 for pitch-bends) in the data fields at the top of the screen. Only controllers of the type and channel you have specified will be varied; the rest will be protected. If no channel is specified, all channels will be processed.

Values allowed for controllers are from 0 to 127, except for pitch-bends, which can go from -8192 to 8191. If you enter a value outside the 0 to 127 range for a controller, the program will not correct you, but when the controller events are varied, they will be set to 0 or 127.

Protection & Restrictions

Vary Controllers presets have their own simplified **PROTECTION** and **RESTRICTIONS** parameters. These are much like the ones found in the other types of presets. Protection and restrictions will operate only if the **On** button beneath the section is highlighted.

Protection is used to specify events that are not to be varied by the preset. The **First** and **Last** parameters indicate the number of events at the beginning and end of the sequence that are not to be varied; notes and other controllers than the ones being varied are counted in determining where protection is to be applied. The **Low** and **High** parameters are used to indicate starting values for controllers (that is, values in the sequence prior to the creation of the variation) above and below which no changes are to be made. For example, if you enter a low value of 10 and a high value of 64, then controller events with values between 10 and 64 will not be varied by the preset.

Switching on the **Reverse** button causes protection to be applied backwards: Only events before the **First** and after the **Last**, or those within the **Low** and **High**, will be protected.

The **RESTRICTIONS** parameters interact only with the **Single** change procedures, not with the global ones. The **Low** and **High** values define a range that can be created by the preset. If a change requested by the **Single** procedure would take the controller value outside the specified range, the change is not made.

Global Processes

The parameters on the upper left quarter of the screen are used to make changes in all controllers (of the specified type and channel) within the sequence. The **WGT** (weight) and **AMT** (amount) designations operate exactly as in presets of the Vary Notes type. A weight creates a likelihood or certainty that a change of the specified type will occur. The effect of the amount parameter depends on the operation being requested.

A **Set** operation will set all unprotected controllers to the value given by the amount parameter. A **Transpose** operation will move the values of all controller events up or down by the specified amount. An **Invert** operation will replace higher values with lower ones and vice-versa, around the unmoving center value specified by the amount parameter. To replace sharp pitch-bends with flat ones, for example, an invert weight of 1 (with 0 **Changes per Vary**) and an amount (center) of 0 should be entered. More unusual bends can be generated by selecting something other than 0 as the center point of the inversion.

A **Scale** operation causes the values of the controller events to be compressed or expanded by some factor. Here, the amount parameter is a percentage: An amount of 100 produces no change, an amount of 200 doubles all controller values, an amount of 50 cuts them in half, and so on. A weight on the **Delete** line will replace controller events of the selected type with DE (delete) events.

Global operations are useful for editing the controller data in a sequence, especially in conjunction with protection of **High** and **Low** range limits.

Single Processes

Again, these work with controller values precisely the way Vary Notes presets work with pitches, durations, and so on. Three slots are provided for **Set, Change, Signed, Gaussian,** and **Scale** operations, so that you can move values up or down by different amounts in a single variation. One controller event will be changed for each weight unit entered.

A **Set** operation causes the value of some controller event to be set to the value specified under **AMT**. A **Change** operation causes the value of some controller event to be moved up or down by the amount given. A **Signed** operation works just like a **Change**, except that a negative amount will always produce a downward change, and a positive amount an upward change. A **Gaussian** operation produces a change whose size is unpredictable. The value entered under amount will be the approximate mean value of the changes created. A **Scale** operation causes the value of some controller event to be multiplied by the percentage entered under **AMT**, where an amount of 100 produces no change.

A weight entered next to Invert will cause the value of some controller event to be flipped with respect to its possible range: CC events and AT events will invert around a center value of 64, while PB events will invert around a center value of 0. The **Delete** parameter will replace controller events of the selected type with DE events. A **Swap** operation will choose two controller events and copy the value of each to the other, while a

Copy operation will copy the value of one event to another.

The musical effect of Vary Controllers presets will depend entirely on what your synthesizer is equipped to do when it receives MIDI controller data. With an instrument like the Oberheim Xpander, which offers a wide variety of MIDI modulation routings, you could insert some controller messages for various controllers at the start of every note using an Add Controllers preset, and then vary the values of these from note to note to produce some highly variable note shapes. The same type of thing can be done on a slightly more restrained scale with Yamaha DX instruments, by using controllers 1 (mod wheel), 2 (breath controller), and -1 (after-touch), and programming some patches to respond to these inputs in different ways.

Chapter 16

Get, Store, Load, & Save

These menu items are used to handle the PVG presets you create. The program holds 80 different presets in memory at a time - 20 each of the Vary Notes, Ornaments, Add Controllers, and Vary Controllers types. Each preset contains a data list of all the parameters you have entered. A Vary Notes preset contains the values and settings on the **Changes**, **Swap/Copy**, **Set Values**, **Global 1**, **Global 2**, and **Split/Pattern** screens, the **GENERAL OPTIONS**, **RESTRICTIONS**, and **PROTECTION** sub-screens, as well as the settings in the **Defaults** window. Presets of the other three types contain only the values and settings shown on their single screens, including the **RESTRICTIONS** and **PROTECTION** sub-screens. The PVG also holds 20 macros, which contain all the settings on the **Macros** screen. When you click on **Get** or **Store** from a single preset screen, the program understands that you want to get or store a preset of that type. When you do so from the **Macros** screen, the program understands that you want to get or store a macro.

After creating or editing a preset or macro, be sure to name it (by typing something over the word **Default** in the lower right corner) before clicking on **Store**. This will make it much easier to find in the preset list.

When you click on **Get** or **Store**, a pop-up window will show you a list of the presets of that type or macros that are currently in memory. To store a preset or macro after making edits in it, point the cursor at the location where you would like to store it and single-click with the left mouse button. (Note that the presets and macros are always in memory, like synthesizer patches. There is no need to store one after getting it and using it, the way you would put a cheese grater back into a kitchen drawer after using it. You should only store it if you wish to save the edits you have made.)

The **Get** operation loads a preset or macro from memory into the currently active buffer. This action erases whatever is in the buffer, so don't use **Get** unless you're willing to lose the contents of the buffer! If you change your mind about the **Get** or **Store** operation, simply click on **CANCEL** at the bottom of the window, and no memory shuffling will be done.

The **Save** operation allows you to save the entire contents of the PVG's memory (all 80 presets and 20 macros) to disk in a file with a .VRY extension. The **Load** operation lets you load such a file from disk. These operations use a standard file select box.

When booted, the KCS will load a file named DEFAULT.VRY if it finds one on the disk. The PVG is shipped with a variety of presets in this file that you might find useful. If you want to, you can rename this file as something else and create your own DEFAULT.VRY file.

Chapter 17

Macros

Single presets in the PVG are designed to be applied in two ways - either to randomly selected notes (or groups of notes), or globally, to all unprotected notes in a sequence. The purpose of macros is to allow you to apply presets more selectively, to precisely defined regions within a sequence, and to use more than one preset in creating a given variation. You can make much more complex edits and variations with a macro than with a simple preset.

In a macro, you can specify a series of presets (up to 16 of them) that are to be used in each edit operation, or to create each variation. These presets can be of any type. You can have different presets operate on different parts of the sequence, perhaps varying the velocities of notes at the beginning (either deterministically or randomly) and adding ornaments at the end. You can define otherwise identical presets to contain different protection settings, and then apply all of them to the entire sequence, which might (for example) have the effect of making a variation in pitch more likely for notes with low velocities than for notes with high velocities. These are only two examples from among hundreds of possibilities.

The most powerful editing feature on the **Macros** screen is probably the **Cyc** (cycle) column. This allows you to apply a preset repeatedly to different parts of a sequence that are separated by a constant value - for example, putting an accent on the downbeat of every bar, or transposing three out of every eight notes while also giving them a new MIDI channel and a fixed velocity.

Get and **Store** work the same way with macros that they do with presets. The PVG can store up to 20 macros, each of which can use the 80 presets in any combination. Using **Load** and **Save** for disk operations loads or saves both presets and macros in a single operation.

The Preset List

The column at the left side of the **Macros** screen shows the names of the presets that are currently in memory. To switch from one type of preset to another, click on either of the double arrow icons at the top of the column. Between the arrows you'll see a word that indicates which set of presets you're currently displaying. Down the center of the screen is a list of the presets that are currently being used in the macro.

When you first call up the **Macros** screen, the list in the center will be blank. If you would like to see what a macro looks like, simply click on **Get** in the menu and select one of the macros that are shipped with the PVG. We have provided some macros that you might find useful. These use some of the factory presets in various ways.

To add a preset to the bottom of the macro list, simply click on its name in the column at the left. Another way to add a preset to the list is to click on the column of two-digit number fields (green underlines) directly below the word **Preset** and then enter the numbers of the presets that you want. Their names will not appear in the center column until you do something that causes the screen to be refreshed, but they will be there even though you don't see them.

If you would like to delete a preset from the list, click on its number (the green number at the left side of the central list, not the white number at the far left) and either type over it with the space bar or press the **Clr/Home** key on the computer. The name will still appear in the list until the screen is refreshed, but the preset will be gone. You can verify this for yourself by blanking out all of the numbers in the list and then clicking on **OK**. The computer will issue an error message: "No lines in Macro!" (Press any computer key after the error message to return to the edit screen.)

A single preset can be used any number of times in a macro. Each time you add a preset, it will be inserted at the earliest possible position in the list. If you have previously deleted a preset from the middle of the list, the new preset will be added in the hole you have created. There is no utility command for inserting a new preset into the middle of an unbroken list, however. If there are no presets that you want to delete in the middle of the list and you would like to insert a new preset somewhere other than at the end, simply copy each of the entries in the list manually onto the next entry until you work your way back to the point where you would like to make the insertion.

A line in the list may also specify that the sequence be processed by the pitch map, which is defined in the Master editor, as described in Chapter 18.Enter 81 under **Preset** to use the pitch map in the line. The **From**, **To**, and **Cyc** switches will have no effect when the pitch map is used in a macro, and the map will be applied to the whole sequence, regardless of any protection settings. If **Ops** is 0 or 1, the sequence will be "run through" the map once, or **Ops** times if **Ops** is greater than 1.

General Options

Along the bottom of the **Macros** screen, just above the exit buttons, is a group of buttons that correspond more or less exactly to the options in the **GENERAL OPTIONS** section of the individual presets. These values override the **GENERAL OPTIONS** section of the individual presets, which have no effect when a macro is being used.

The number under **Variations** is the number of variations to be created by the macro. **Orig, Consec, Evol**, and **Rev** are shorthand for **Overwrite Original, Consecutive Mults, Evolving Mults**, and **Reverse** on the preset screen. (See Chapter 3 for more on these options.) Likewise, the **Edit Mode** switch performs the same function as the **Edit Mode** switch in a preset. It causes one variation to be created that overwrites the original.

Lines

The number following the word **Lines** is the number of macro lines (individual presets with their associated **From**, **To**, and **Cyc** numbers) to be executed by the macro. If this number is 0, each line will be executed once. If it is a positive integer, lines will be selected at random the number of times specified by the **Lines** number. Note that this may cause some lines to be executed more than once, while others are not executed at all.

Con Lines

When this switch is highlighted, the program uses a new consecutive line within the macro for each variation in a series. The total number of series items created (and thus the total number of lines executed) will be determined by the **Variations** parameter in the bottom row.

For some macros, especially in edit mode, it doesn't matter what order the lines are in. Since the **Con Lines** button causes lines to be executed in order, however, the order may make quite a difference. If a note has been changed to a new pitch by line 2, for example, it might fail a logical test that takes place in line 5. If the order of the lines were reversed, the result might be quite different. Another example: If a preset that performs pitch transposition has a PROTECTION value set for notes with high velocities, it makes a big difference whether another preset that increases velocities is executed before or after the first. Consecutive line execution can also be important when evolving mults are being generated.

Range Unit

Three choices are available for the range unit setting: **Event**, **Step**, and **Meas** (measure). This choice determines how the program will interpret the From, To, and Cyc values in the list. The choices correspond precisely to the three values computed by the KCS at the left side of the edit screen event list: MSR (measure), ST (step), and EVNT (event). A quick glance at this screen should give you an immediate understanding of how From, To, and Cyc will be interpreted.

Let's look at a simple example using the values **From** = 1, **To** = 1, **Cyc** = 3, with a preset in the list that globally transposes notes down by an octave. When **Event** is selected as the range unit, the program will look through the source sequence and transpose every third event (because **Cyc** = 3) down an octave - provided, of course, that that event is a note. When **Meas** is selected as the range unit, all of the notes in every third measure (again, because **Cyc** = 3) will be transposed down an octave. When **Step** is selected as the range unit, the program will begin counting its clock steps at the beginning of the sequence, and transpose any note that falls on a clock step divisible by 3.

Ops

Each line in a macro has four parameters associated with it - **From**, **To**, **Cyc**, and **Ops**. The **Ops** (operations) number, in the right-most column, overrides the **Changes per Vary** number in the preset. As with **Changes per Vary**, a 0 setting will cause the program to make one change for each **WGT** (weight) value in the preset, while any other number will cause that number of changes to be generated by the line.

From/To Ranges & Cycles

The **From**, **To**, and **Cyc** parameters replace the **First** and **Last** parameters in the **PROTECTION** section of the individual presets. These two PROTECTION parameters (and only these two) are always ignored when a macro is used.

From and **To** values are used to specify a range of events, measures, or steps within each measure to which the preset in that line will be applied. The effect of **From, To,** and **Cyc** values thus depends on the range unit setting (see above). If you set **From** to 5 and **To** to 11, the preset will be applied to events 5 - 11 of the sequence if **Event** is selected, to measures 5 - 11 if **Meas** is selected, and to any events that fall on clock steps 5 - 11 of each measure if **Step** is selected. The range unit will always be of the same type (**Event, Step,** or **Meas**) for all the lines at any given time.

If the **To** value is left blank, the program will begin executing the preset at the **From** point and proceed to the end of the sequence. Likewise, if a value is entered under **To** but nothing is entered under **From**, the program will assume that **From** is 0. A **From** value can be negative, which can be useful when the range unit is set to **Step**, allowing you to anticipate a beat. (See the example below.)

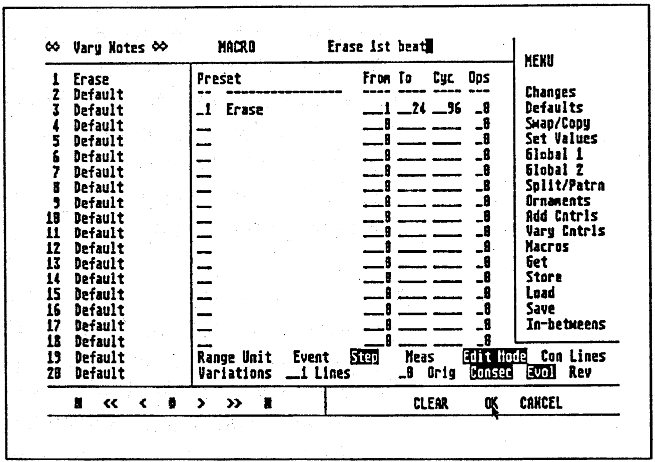
The **Cyc** (cycle) value allows you to specify that a line is to be applied repeatedly to sections of the sequence separated by the **Cyc** value. For example, with a range unit of **Event**, a **From/To** of 3 - 5, and a **Cyc** of 12, the preset will be executed for notes 3 - 5, 15 - 17, 27 - 29, 39 - 41, and so on.

In order for a **Cyc** value to be operative, it must be larger than the **To** value. This prevents two conditions: First, you can't use a macro in such a way that a line operates on overlapping regions. This would happen, for example, with a **From** of 3, a **To** of 9, and a **Cyc** of 2, which (in combination with a range unit of **Event**) would cause the line to operate on events 3 - 9, 5 - 11, 7 - 13, and so on, if that were permitted by the program. And second, you can't start (or end) a cyclic edit in the middle of a sequence. The cycle must be continuous from one end to the other (except that the beginning of the first occurrence of the cycle may fall before the beginning of the sequence, if the **From** value is negative, and the sequence may end in the middle of the last occurrence of the cycle).

If a range was highlighted in the KCS edit screen when vary was selected, the word **RANGE** will appear on the bottom line. Clicking on this will cause the range unit to be set to **Event**, and the **From** and **To** values for each line in the macro to be set to the start and end points of the highlighted range. Unless you enter a **Cyc** value, this will cause all of the lines to operate exclusively on the range.

Cycling can easily be applied creatively to generate patterns that have a natural rhythmic cycle. For example, suppose we want to emphasize all the notes that fall in the first beat of every measure. Assuming that we are using the KCS defaults of 24 steps per beat and 96 steps per measure, we would choose a preset that transposes velocity up (using the **Global 1** screen) or sets it to a high value (using the **Global 2** screen), then enter 1 under **From**, 24 under **To**, and 96 under **Cyc**, and select **Step** as the range unit. If we wanted instead to emphasize notes that fall within three clock steps of any quarter-note, we would set **From** to -2, **To** to 4, and **Cyc** to 24. Emphasizing the first measure in every four-measure phrase would simply be a matter of selecting **Meas** as the range unit and then setting **From**, **To**, and **Cyc** values of 1, 1, and 4.

"Emphasizing" is really too narrow a concept to illustrate the wealth of possibilities. The PVG could be used to add pitch-bends or ornaments to notes within any of the ranges just discussed, or to randomize pitches and destabilize event times, or to make all of these changes at once in overlapping regions. The power of macros extends far beyond our ability to visualize what we might do with them musically.

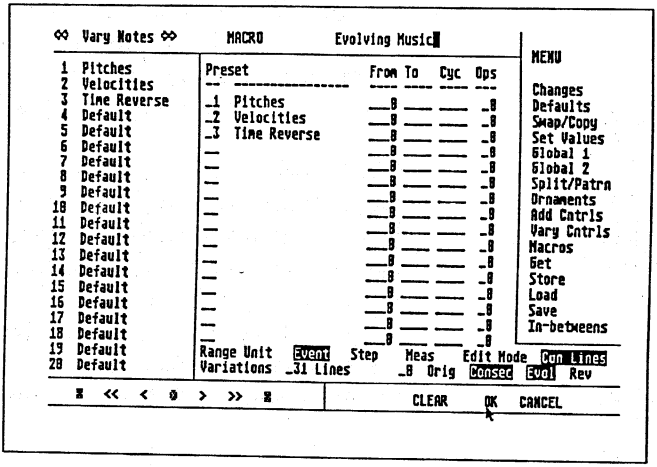


```
⇔ Vary Notes ⇔      MACRO        Erase 1st beat

 1 Erase       Preset                    From To   Cyc  Ops
 2 Default     -- ---------------        ---- ---- ---- ---
 3 Default     _1 Erase                  ___1 __24 __96 _0
 4 Default     __                        ___0 ____ ____ _0
 5 Default     __                        ___0 ____ ____ _0
 6 Default     __                        ___0 ____ ____ _0
 7 Default     __                        ___0 ____ ____ _0
 8 Default     __                        ___0 ____ ____ _0
 9 Default     __                        ___0 ____ ____ _0
10 Default     __                        ___0 ____ ____ _0
11 Default     __                        ___0 ____ ____ _0
12 Default     __                        ___0 ____ ____ _0
13 Default     __                        ___0 ____ ____ _0
14 Default     __                        ___0 ____ ____ _0
15 Default     __                        ___0 ____ ____ _0
16 Default     __                        ___0 ____ ____ _0
17 Default     __                        ___0 ____ ____ _0
18 Default     __                        ___0 ____ ____ _0
19 Default     Range Unit  Event  Step   Meas    Edit Mode  Con Lines
20 Default     Variations  __1 Lines     _0 Orig  Consec  Evol  Rev
```

MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
0 << < 0 > >> 0
CLEAR
OK
CANCEL

Macro to apply the Erase preset to the first beat of every measure of the current sequence. If the Erase preset was programmed to always do a global erase, then the first beat of every measure will be erased.

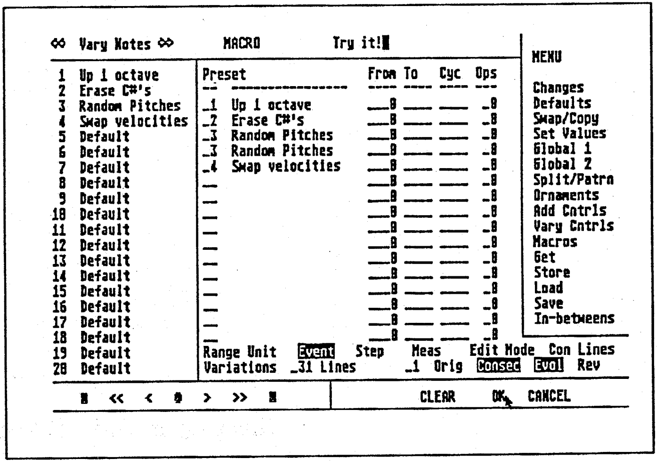


```
⇔ Vary Notes ⇔      MACRO        Evolving Music

 1 Pitches       Preset                  From To   Cyc  Ops
 2 Velocities    -- ---------------      ---- ---- ---- ---
 3 Time Reverse  _1 Pitches              ___0 ____ ____ _0
 4 Default       _2 Velocities           ___0 ____ ____ _0
 5 Default       _3 Time Reverse         ___0 ____ ____ _0
 6 Default       __                      ___0 ____ ____ _0
 7 Default       __                      ___0 ____ ____ _0
 8 Default       __                      ___0 ____ ____ _0
 9 Default       __                      ___0 ____ ____ _0
10 Default       __                      ___0 ____ ____ _0
11 Default       __                      ___0 ____ ____ _0
12 Default       __                      ___0 ____ ____ _0
13 Default       __                      ___0 ____ ____ _0
14 Default       __                      ___0 ____ ____ _0
15 Default       __                      ___0 ____ ____ _0
16 Default       __                      ___0 ____ ____ _0
17 Default       __                      ___0 ____ ____ _0
18 Default       __                      ___0 ____ ____ _0
19 Default       Range Unit  Event  Step   Meas    Edit Mode  Con Lines
20 Default       Variations  _31 Lines     _0 Orig  Consec  Evol  Rev
```

MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
0 << < 0 > >> 0
CLEAR
OK
CANCEL

Macro to produce 31 consecutive evolving variations on the current sequence. The three selected presets are applied in order, i.e. first Pitches, then Velocities, then Time Reverse, then Pitches again ...



```
∞ Vary Notes ∞        MACRO           Try it!█

 1  Up 1 octave       Preset                          From To   Cyc  Ops
 2  Erase C#'s        --  ----------------            ---- ---- ---- ---
 3  Random Pitches    _1  Up 1 octave                 ___0 ____ ____ _0
 4  Swap velocities   _2  Erase C#'s                  ___0 ____ ____ _0
 5  Default           _3  Random Pitches              ___0 ____ ____ _0
 6  Default           _3  Random Pitches              ___0 ____ ____ _0
 7  Default           _4  Swap velocities             ___0 ____ ____ _0
 8  Default           __                              ___0 ____ ____ _0
 9  Default           __                              ___0 ____ ____ _0
10  Default           __                              ___0 ____ ____ _0
11  Default           __                              ___0 ____ ____ _0
12  Default           __                              ___0 ____ ____ _0
13  Default           __                              ___0 ____ ____ _0
14  Default           __                              ___0 ____ ____ _0
15  Default           __                              ___0 ____ ____ _0
16  Default           __                              ___0 ____ ____ _0
17  Default           __                              ___0 ____ ____ _0
18  Default           __                              ___0 ____ ____ _0
19  Default           Range Unit  Event  Step   Meas    Edit Mode  Con Lines
20  Default           Variations  _31 Lines      _1  Orig  Consec  Evol  Rev
```

MENU
Changes
Defaults
Swap/Copy
Set Values
Global 1
Global 2
Split/Patrn
Ornaments
Add Cntrls
Vary Cntrls
Macros
Get
Store
Load
Save
In-betweens
■ << < ● > >> ■
CLEAR
OK
CANCEL

Preset to create a series of evolving variations. For each variation one of the selected presets is picked at random. The Random Pitches preset is twice as likely to be picked as the other. If lines were set to 0, all of the presets would be used for each variation, and Random Pitches would be used twice.

This page intentionally left blank.

Chapter 18

Master Editor

When you click on **Master Edit**, in the first column of the edit options section of the KCS edit screen, you will enter the Master Editor. The Master Editor allows you to edit the current sequence in a number of ways. There are six "pages" within the Master Editor. To move from one page to another, simply click on an item in the menu list in the upper right corner of any page.

Most of the operations in the Master Editor work just like those in the PVG and the rest of the KCS: You click on switches to highlight them and enter the values that you want, and then click on **OK**. If you click on **CANCEL** instead, you will be taken back to the edit screen and no change will be made in the current track or sequence. If you're not clear how to use the Master Editor's controls, consult the KCS manual.

One important difference between the Master Editor and the PVG is that only one page of the Master Editor will have an effect during any given edit operation. There's no need to clear values in the other pages in order to avoid, for example, deflamming chords while blending a rhythm. Only the values that you can currently see will be used during the edit operation. You can perform as many operations at once as you'd like, as long as they're on a single page.

Edit Range

The first four pages of the Master Editor all have data fields for an edit range displayed at the top. This will default to **All** when you enter a new page, unless you have highlighted a range on the edit screen, in which case it will default to **Event Range** with the numbers of the events at the boundaries of the range loaded into the start point and end point data fields to the right of the highlighted button. If you want to edit only a limited range rather than all events, you must click on the appropriate button (if it is not already highlighted) and then enter values in the start and end fields (unless the default values are the ones you want).

Blend

The operations on the **Blend** page allow you to combine the data in the currently selected track or sequence with the contents of the Open mode sequence that is selected by the number to the right of the words **REFERENCE SEQUENCE.** This sequence is not affected by the operation, so the "blending" is not quite the same as what you'd get if you actually put the two sequences into a blender. We will call the unaffected sequence selected on this line the reference sequence. The sequence that will be affected is, of course, the sequence that was on the edit screen when the Master Editor was entered.

The blending can take place in various ways, depending on what options you select. The most straightforward type of blending occurs when one or more of the **Match** buttons at the bottom of the screen are highlighted. Each button will replace the data in the current sequence with the indicated data in the reference sequence. If you select several of the buttons, the current sequence will resemble the reference sequence more closely after the edit operation.

In a **Match** operation, the events are correlated on a one-to-one basis, as if the two sequences were printed out side by side in two columns on a sheet of paper, and the selected values were copied from the first column to the corresponding entry in the second column. If the reference sequence contains DE events (other than the final DE event, which is ignored except when matching rhythm), the **Match** operation will change the corresponding events in the current sequence to DE events. If other types of non-note events are present in the reference sequence, the result of the operation will be unpredictable.

If the reference sequence is shorter than the sequence being edited, the program will start over again at the top of the reference sequence when it reaches the bottom. This fact is extremely useful if you want a short repetitive pattern to be created by the edit. For example, if you have a monophonic line in the current sequence, and you want to edit it so that it will be played with a shuffle rhythm (such as a pattern of 14 clocks, 10 clocks, 14, 10, 14, 10, and so on), there's no need to create a reference sequence the same length as the sequence that you want to edit. Simply create a reference sequence with two notes in it, in the proper rhythm (in this case, times of 0 and 14 for the two notes, with a final DE event that has a time of 10).

The operations selected in the middle of the screen differ from the **Match** operations in that they correlate with the absolute times in the reference sequence rather than with the event number. For example, let's say that your reference sequence contains a monophonic C major scale in half-notes, starting on C, and that your current sequence contains a lot of sixteenth notes of various pitches. If you select one of the **Pitch Follow** options, all of the sixteenth notes within the first two beats of the current sequence will follow the first pitch (the C) in the reference sequence.

Auto-correct aligns the rhythmic values in the current sequence with those in the reference sequence - a much more flexible type of auto-correction than the standard type found in the KCS edit screen, which auto-corrects only to multiples of a single rhythmic value. Master Editor auto-correcting could be useful if you've improvised a free-flowing lead and would like selected notes within it to be accented by notes on a second track: There's no need to play the accent notes with perfect accuracy, or to haul out the calculator and edit them by hand. Simply hit them as closely as you can and then auto-correct using the Master Editor.

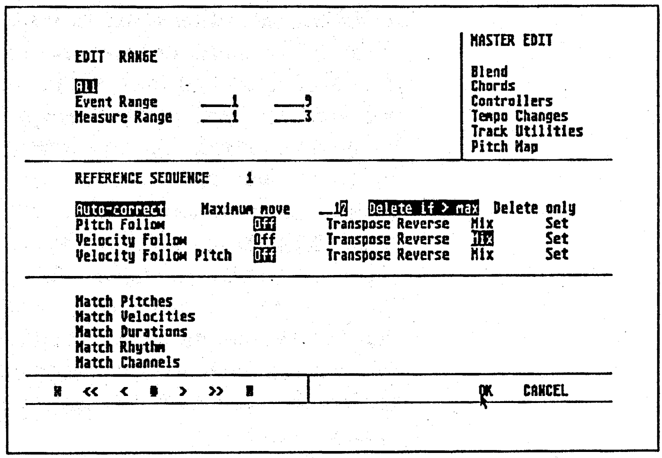
If a number is entered in the data field to the right of **Maximum move**, no auto-correction will be applied to an event if it would have to be moved farther than this value. If you click on **Delete if > max**, the program will delete all of the notes that are too far off to auto-correct (depending on the **Maximum move** value you have entered) by replacing them with DE events.

When the **Delete only** button is highlighted along with the **Auto-correct** and **Delete if > max** buttons, the program will delete any notes that are too far away to auto-correct, but it will not actually perform the auto-correction on the notes that remain. This is a way of deleting any notes on a track that fall further than a certain time limit from notes on the reference track. If the reference track contains whole-notes in 4/4, only the notes on the current track that are clustered around the downbeats will remain after the **Delete only** operation. (The width of the cluster will depend on the **Maximum move** parameter.)

If an option in the **Pitch Follow** row is selected, the pitches in the current sequence will be edited with reference to the pitches in the reference sequence. Likewise, if a **Velocity Follow** option is selected, velocities will be edited with reference to velocities. The **Velocity Follow Pitch** selections will cause the velocities in the current sequence to be edited with reference to the pitches in the reference sequence: Note number 24 (C1) will be considered equivalent to a velocity of 24, and so on.

To the right of each of the follow options is a series of radio buttons. (Like the buttons on a car radio, only one will be active at a time.) **Set** will replace the values in the current sequence with those in the reference sequence. **Mix** will produce an average of the two values. **Transpose** will figure out how far the values in the reference sequence are from Middle C (C4 - MIDI note 60) or a velocity of 64, and will move the values in the current sequence up or down by this same amount. **Reverse** will do an upside-down transpose operation, in which the values in the current sequence will be moved up if those in the reference sequence are below the median point, and vice-versa.

The **Auto-correct** feature can be used (possibly along with **Velocity Follow**) to create complex syncopated patterns. If we set up a reference sequence it two notes and times of 0 and 7, with 5 for the final DE event, and autocorrect to it we will be autocorrecting to sixteenth notes, but each pair of sixteenth notes will be "swung" within its eighth note. If the two notes in the reference sequence have different velocities, we can accent either sixteenth note within an eighth note relative to the other. A reference sequence with six 12's followed by twenty-four 1's in the TIME column would autocorrect the first three beats of a measure to eighth notes, while leaving the last measure unaltered.



```
                                                 MASTER EDIT
 EDIT  RANGE
                                                 Blend
 All                                             Chords
 Event Range        ___1      ___9               Controllers
 Measure Range      ___1      ___3               Tempo Changes
                                                 Track Utilities
                                                 Pitch Map

 REFERENCE SEQUENCE    1

 Auto-correct     Maximum move      __12  Delete if > max  Delete only
 Pitch Follow               Off     Transpose Reverse  Mix      Set
 Velocity Follow            Off     Transpose Reverse  Mix      Set
 Velocity Follow Pitch      Off     Transpose Reverse  Mix      Set

 Match Pitches
 Match Velocities
 Match Durations
 Match Rhythm
 Match Channels

 ■  <<  <  ■  >  >>  ■                          OK    CANCEL
```

Master Editor setting to auto-correct the current sequence to the rhythm pattern of sequence 1. Each notes' velocity will be averaged with the velocity of the corresponding note in sequence 1. Notes more than 12 time steps away from a "hit" will be deleted.

Chords

Operations on this page will be applied only to chords (that is, stacks of events in which all but the first event has a time of 0) within the sequence, and ignore all other notes.

But what if you've played some chords a little sloppily, so that some of the times are 1 or 2 rather than 0? In order for the program to know that you meant this as a chord, you should begin the edit by entering a value under **Deflam**. The program will look for events that are within this number of clocks of the first event in each chord, and will move these events backward so that their time is 0. (The time of the next event will be adjusted to compensate.) The deflam value must be smaller than the time between pairs of events that you don't want turned into rhythmic unisons.

A flam, incidentally, is a type of drum hit in which both sticks hit the same drum at nearly, but not quite, the same moment. So deflamming means aligning events to exactly the same time value.

A **Deflam** operation looks forward only from the first event in the chord, or from the first of the events that it is aligning so that they will become a chord. It does not look forward from subsequent events that are being deflammed. To see how this works, enter a series of eight or ten notes that are separated by 1 clock step, and enter a deflam value of 3. If each event were deflammed with reference to any preceding event, all of the events in your test sequence would stack up into a single chord. You will find, however, that the **Deflam** operation produces a group of four-note chords, each of which is separated by 4 steps (one greater than the largest length of time allowed by the deflam value).

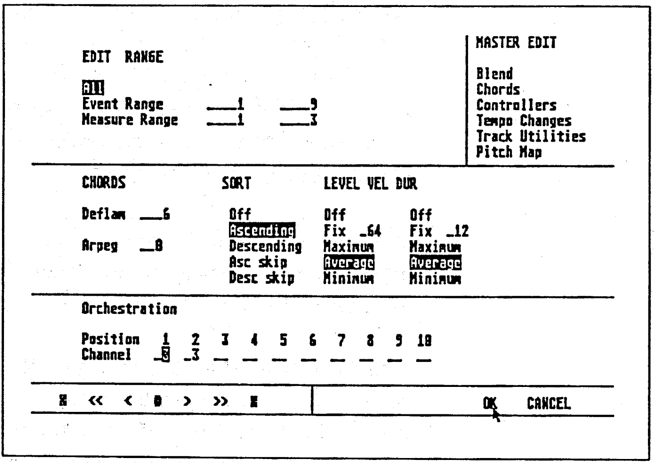
Arpeg (arpeggiation) is used to spread the notes in a chord out across time. Instead of having times of 0, each note will have the time value entered under **Arpeg**. Any notes that the arpeggiation operation would move past the next event in the sequence will be replaced by DE events.

The options under **Sort** allow you to change the order of the notes in a chord by pitch. **Ascending** and **Descending** do exactly what you would expect. The highest or lowest note in a chord is the first event, and the others follow in order. **Asc skip** (ascending skip) and **Desc skip** (descending skip) produce an ascending or descending order in which every other note is skipped until the other extreme is reached, after which the skipping process moves back toward where it started, picking up the notes it skipped the first time. For example, if a chord consists of the notes A, B, C, D, E, F, and G, an ascending skip operation will order them A, C, E, G, F, D, B, while a descending skip will order them G, E, C, A, B, D, F.

Sorting operations are often helpful prior to arpeggiation. Sorting insures that the elements of the arpeggio will be in the same order for each chord.

The two columns to the right of **Sort** are used for editing the velocity and duration of notes within a chord. To set all values to a fixed level, click on **Fix** and enter the value you want. Selecting **Maximum, Average**, or **Minimum**, will set all the velocities or durations within each single chord to be equal to the highest or lowest value, or to an average of the values. The comparing and averaging operations are done for only one chord at a time, so if you play a short chord and a long chord in the same sequence and ask for duration averaging, the short chord will still be short and the long chord will still be long.

The fields under **Orchestration** are used to set the MIDI channel of chord notes with reference to the position (the numerical order of the events in the event list) of the notes in the chord. Again, using a **Sort** operation before changing the channels will cause the chords in a passage to have a more unified sound: Each element in a chord will always go to the same MIDI channel. And as with other chord operations, non-chord notes in the sequence will not be affected by the **Orchestration** operation.



EDIT RANGE

All
Event Range 1 9
Measure Range 1 3

CHORDS

Deflam 6
Arpeg 0

SORT: Ascending

LEVEL VEL: Fix 64, Average
LEVEL DUR: Fix 12, Average

Orchestration

Position 1 2 3 4 5 6 7 8 9 10
Channel 3 3

MASTER EDIT
Blend
Chords
Controllers
Tempo Changes
Track Utilities
Pitch Map
Off
Descending
Asc skip
Desc skip
Off
Maximum
Minimum
Off
Maximum
Minimum
<< < > >>
OK
CANCEL

Setting to deflam all chords with a width of 6 time steps or less. The velocity and duration of each chord will be set to the average of the notes within it. Chords will be sorted in ascending order, and the first two notes of each chord will be placed on channel 3.

Controllers

This page allows you to perform several useful operations to the non-note MIDI events recorded by the KCS. The type of event to be operated on is selected in the column on the left. If you are editing continuous controllers (CC events), you must specify the number of the controller in the data fields to the right of the words **Control Changes**.

Three operations can be performed on the non-note events. **Split** copies them to the next available empty sequence, and erases them from the current sequence. **Erase** erases them without copying them (and, obviously, without affecting any other type of data in the sequence). **Thin** is a more complex operation which operates somewhat like running a comb through the selected data type, removing some of the events (corresponding to the teeth of the comb) while leaving others (the gaps between the teeth) untouched. Thinning cannot be executed with program changes or polyphonic key pressure.

The number entered under **Fraction Kept** determines how much thinning is done. If the **Fraction Kept** value is 2, then 1/2 of the events will be kept; if it is 3, then 1/3 will be kept; if it is 4, then 1/4 will be kept; and so on. The last controller event before a note event will not be erased, nor will a zero value or a minimum or maximum pitch-bend value. If a controller event is more than the **Maximum Time** value away from the previous event of that type, it will not be thinned.

The thinning operation will not work correctly in a multi-channel sequence.

Thinning can be useful when a track is so full of control changes that the MIDI data output is being slowed down. It is often possible to thin the data by a factor of 2 or more without introducing any audible change in the expressivity of the performance. Larger thinning values could be used for special effects, such as introducing some irregular microtonal staircasing into a pitch-bend. (For best results with this effect, try a wide pitch-bend range setting on the synthesizer.)

Tempo Changes

The options on this page allow TM (tempo) events to be erased or scaled up or down by a percentage. When **Scale** is selected, the number to its right is the percentage by which all TM values are multiplied (100 = no change).

If you have entered AC (accelerando) or DC (decelerando) events into a track, you can use the **Convert AC/DC Events** button to change these events into TM events - in other words, to give them absolute rather than relative tempo values. The conversion is not reversible. Converting to TM events could be useful if your song has a number of tempo changes that you have entered with AC and DC events, and you find that all of the sections are now at the tempo you want except for one in the middle. Changing the AC or DC event at the start of this section will change the tempo of all subsequent sections as well, because AC and DC tempos are computed as a percentage of whatever the current tempo happens to be. When the AC/DC events are converted to TM events, you can edit one TM event without affecting later ones.

Track Utilities

This page is a convenient shortcut for making some basic changes in the shape of an entire arrangement. As the screen cautions, changes made here will affect all tracks, and cannot be undone with the undo command. The **Track Utilities** page will have no effect on sequences in Open mode - but if you have material in memory in both Track mode and Open mode, be careful! Doing an **Erase, Delete,** or **Insert** on this page while you are working in Open mode will affect the material in the Track mode area.

The **Delete range** and **Erase range** commands use the same terminology as the rest of the KCS: Deleting a range will cause it to be closed up, so that events after the range will be earlier. Erasing a range will leave a gap: The times of subsequent events will not be affected, because a time value the same length as the gap will be added to the next event on each track. The boundaries of the range to be erased or deleted must be entered under range start and range end.

Insert space creates a new blank spot on all tracks, beginning at the time entered under **Range start** and lasting for the number of measures and steps entered under **Insert amount.** A time value equal to the size of the gap will be added to the event on each track that falls next after the beginning of the gap.

Inserting a space can be useful if you have recorded a verse on several tracks and later want to go back and scissor in an intro, or if you have recorded a verse and chorus and then find that you need an extra bar between the two. Likewise, deleting a range from all tracks will tighten up an arrangement that has extra bars in it.

Pitch Map

The **Pitch Map** page is used for; changing all occurrences of any MIDI note in a sequence into any other MIDI note. This would be useful, for example, if a drum track had been recorded into the sequencer to be played by one drum machine, and it was later desired to fire a different drum machine, one that assigned its sounds to different MIDI notes, from the same track. In addition to remapping pitches, the pitch map allows you to assign new velocities and MIDI channels to the notes, or to erase selected notes.

While MIDI defines 128 notes, the pitch map can be used to edit only 88 notes, from C0 (note 12) through D#7 (note 99). If you should need to use pitch mapping to edit notes that fall outside this range, simply begin by transposing the entire sequence into the allowed range. The possible target pitches for remapping extend up through G9 (note 127). If for some reason you should need to remap to MIDI notes 1 through 11, you can do a remapping to the next octave above the target pitches, and then transpose the sequence down globally after remapping.

The white note names to the left of the columns serve to mark the original pitches that the data fields will alter. If you like, you can imagine entire columns of white pitch names - C0, C#0, D0, D#0, and so on. Only the C's and F's are displayed in order to increase visual clarity. The white note names (including the invisible ones) are the original notes that will be altered. The green note names are the targets: the notes that will be produced by the mapping operation. Initially the pitch map defaults to a mapping of every pitch onto itself.

To the right of the pitch names are two columns, one for velocity and one for MIDI channel. The meaning of the entry in the velocity column depends on which flag is highlighted in the column at right, beneath the word **VELOCITY.**

If **Set** is highlighted, the velocity value of each note in the sequence will be set to the velocity value indicated. If a velocity field is set to 0, the velocity of that note won't be changed.

If **Mix** is highlighted, the program will generate an average of the current velocity value and the velocity value shown in the pitch map. And if **Transpose** is highlighted, the program will compute the difference between the velocity shown in the pitch map and a center value of 64, and add or subtract this value from the note(s) in the sequence. For example, if a note's current velocity is 96, and a value of 50 is entered in the velocity column for that note, the velocity produced will be 82, because 50 - 64 = -14, and 96 + -14 = 82.

The right data field in each column is used to set the MIDI channel. This operation will only take place if **Channels/Deletes** is highlighted. Leaving the field blank will cause the channel(s) of notes with that pitch to be unchanged. Entering a channel of 0 will cause notes with that pitch to be deleted from the sequence.

If the **Recursive** button is highlighted, the pitch map will be applied to the original sequence and the result appended to that sequence. The process will be repeated for the number of times specified by the value to the right of this switch. This feature was added to make it easier to create the musical equivalent of Mandelbrot drawings, as discussed in the Preface.

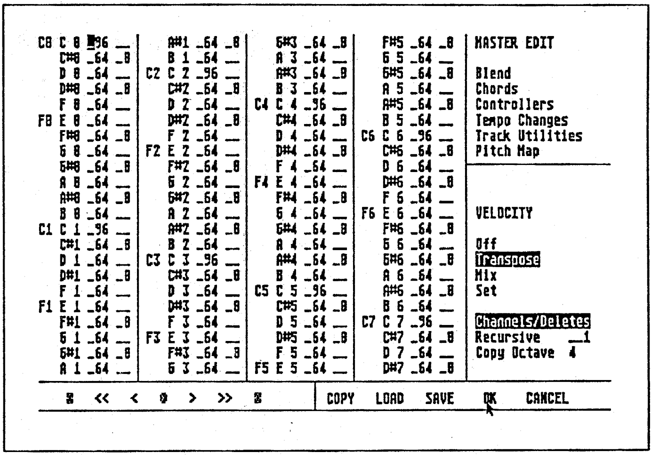
The **Load** and **Save** buttons on the Pitch Map screen are used to access .VRY files, which store the pitch map in addition to the presets and macros.

The **COPY** button is used to map the current sequence onto the pitch map. Each note in the sequence replaces the following note in the sequence, in the pitch map. For example, if you copy a three note sequence C4, E4, F5 into the pitch map, the E4 slot in the pitch map will be filled with C4, E4 will be placed in the F5 position, and F5 will replace the value in the C4 location. No other notes in the pitch map will be changed.

Pitch mapping can be useful for altering notes that are wrong for the harmonic context of the sequence, or for changing from one melodic mode to another. These edit operations can also be performed using global presets in the Programmable Variations Generator.

The difference is that the PVG will always treat notes with the same letter name as identical, no matter what octave they are in. Pitch mapping allows you to treat each octave in its own way.

You could also use pitch mapping very conveniently when composing music for a microtonal synthesizer. A piece could be recorded initially from a standard keyboard in standard 12-tones-per-octave tuning, and then could be remapped so that the MIDI notes play the correct "keys" of the microtonal instrument.



```
C0 C 0 _96 __ |    A#1 _64 _0 |    G#3 _64 _0 |    F#5 _64 _0 | MASTER EDIT
   C#0 _64 _0 |    B 1 _64 __ |    A 3 _64 __ |    G 5 _64 __ |
   D 0 _64 __ | C2 C 2 _96 __ |    A#3 _64 _0 |    G#5 _64 _0 | Blend
   D#0 _64 _0 |    C#2 _64 _0 |    B 3 _64 __ |    A 5 _64 __ | Chords
   F 0 _64 __ |    D 2 _64 __ | C4 C 4 _96 __ |    A#5 _64 _0 | Controllers
F0 E 0 _64 __ |    D#2 _64 _0 |    C#4 _64 _0 |    B 5 _64 __ | Tempo Changes
   F#0 _64 _0 |    F 2 _64 __ |    D 4 _64 __ | C6 C 6 _96 __ | Track Utilities
   G 0 _64 __ | F2 E 2 _64 __ |    D#4 _64 _0 |    C#6 _64 _0 | Pitch Map
   G#0 _64 _0 |    F#2 _64 _0 |    F 4 _64 __ |    D 6 _64 __ |
   A 0 _64 __ |    G 2 _64 __ | F4 E 4 _64 __ |    D#6 _64 _0 |
   A#0 _64 _0 |    G#2 _64 _0 |    F#4 _64 _0 |    F 6 _64 __ |
   B 0 _64 __ |    A 2 _64 __ |    G 4 _64 __ | F6 E 6 _64 __ | VELOCITY
C1 C 1 _96 __ |    A#2 _64 _0 |    G#4 _64 _0 |    F#6 _64 _0 |
   C#1 _64 _0 |    B 2 _64 __ |    A 4 _64 __ |    G 6 _64 __ | Off
   D 1 _64 __ | C3 C 3 _96 __ |    A#4 _64 _0 |    G#6 _64 _0 | Transpose
   D#1 _64 _0 |    C#3 _64 _0 |    B 4 _64 __ |    A 6 _64 __ | Mix
   F 1 _64 __ |    D 3 _64 __ | C5 C 5 _96 __ |    A#6 _64 _0 | Set
F1 E 1 _64 __ |    D#3 _64 _0 |    C#5 _64 _0 |    B 6 _64 __ |
   F#1 _64 _0 |    F 3 _64 __ |    D 5 _64 __ | C7 C 7 _96 __ | Channels/Deletes
   G 1 _64 __ | F3 E 3 _64 __ |    D#5 _64 _0 |    C#7 _64 _0 | Recursive    __1
   G#1 _64 _0 |    F#3 _64 _0 |    F 5 _64 __ |    D 7 _64 __ | Copy Octave  4
   A 1 _64 __ |    G 3 _64 __ | F5 E 5 _64 __ |    D#7 _64 _0 |
```

Pitch Map setting to exchange all E's and F's, delete all sharps, and add 32 to the velocity of all C's.

This page intentionally left blank.

Chapter 19

Polyphonic Key Pressure

The Level II KCS allows recording, playback and editing of polyphonic key pressure events. While poly key pressure is not a recent addition to the MIDI spec, it has been almost universally ignored until recently. Two instruments recently introduced by Ensoniq send an independent key pressure value for each note being held down, and the Kurzweil MIDIboard and the late Prophet T8 from Sequential also send polyphonic key pressure. Recording of poly key pressure is enabled by turning on the **AFTOUCH** button on the play/record screen or the **Set Options** page. On the edit screen, polyphonic key pressure events have a type of PK, followed by a MIDI note value in the note column and a key pressure value in the velocity column.

MSR-	ST	EVNT	TIME	CH	TYP	NOTE	VEL	DUR
1-	1	1	0	1	PK	C 3	0	
1-	1	2	0	1	PK	C 3	1	
1-	1	3	0	1	PK	C 3	3	
1-	2	4	1	1	PK	C 3	5	
1-	2	5	0	1	PK	C 3	7	
1-	3	6	1	1	PK	C 3	12	
1-	3	7	0	1	PK	C 3	18	
1-	4	8	1	1	PK	C 3	22	
1-	5	9	1	1	PK	C 3	29	

Individual PK events are edited in the same manner as note off events. The only bulk operations affecting PK events are **Erase Controllers** from the Transpose page, and the **Split** or **Erase** options from the **Controllers** page of the Master Editor.

This page intentionally left blank.

Index

Add Controllers screen 17, 75
AMT 37
Arithmetic strip 14
Auto-correction 33
    of in-betweens 66
    with Blend 101
Blend 100
CANCEL 22
Changes
    Constant 39
    Gaussian 39
    global 53
    interval 42
    Signed 39
Changes per Vary 21, 37
Changes screen 13, 37, 42
Chords 103
CLEAR 12, 22
Consecutive Mults 19, 66
Controllers
    global changes 84
    single changes 85
    splitting, erasing, and thinning 106
Defaults 35, 36
Disk operations 87, 110
Edit Mode 21
Evolving Mults 20
Exit buttons 22
GENERAL OPTIONS 15, 19, 23
Get 88
Global 1 screen 16, 54, 55
Global 2 screen 16, 55, 57
In-betweens 11, 65, 66
Interval patterns 60
Macros 13, 87, 89, 95
    cycles 93
Master Editor 99, 111
Menu 14
Modulation
    of controller events 78
    of ornaments 72
Ornaments screen 16, 67, 73

Overwrite Original 19
Pitch map 91, 109
PK events 113
Presets 12, 87
    in macros 90
PROCESS CONSECUTIVE 45
Program changes 77
Protection 15, 25, 29
    controllers 83
    duration range 28
    First and Last 27
    intervals 28, 29
    pattern testing 59
    pitch range 27
    probability 26
    reverse 26
    time range 28
    velocity range 28
RANGE 22, 95
Restrictions 15, 31, 34
    controllers 83
    defaults 15, 31, 35
    duration 34
    pitch range 32, 35
    velocity range 32, 35
Reverse 20
Scale positions
    of in-betweens 66
    protecting 26
    restricting 31, 36
SD 35, 39
Set Values
    delete and erase 50
    global 56
    interval 50
Set Values screen 16, 49
Split/Pattern screen 16, 59
Staccato/Legato 42
Standard deviation 35, 39
Store 88
Swap 43
Swap/Copy screen 16, 43, 48
Tempo Changes 107
Track mode 11
Track Utilities 108

Undo 19
Variations 19
Vary Controllers screen 17, 83
WGT 37, 43
Xtime 47

This page intentionally left blank.